Journal of Biomedical Informatics 51 (2014) 219–241

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin

A general framework for time series data mining based on event analysis: Application to the medical domains of electroencephalography and stabilometry

Juan A. Lara [a,*], David Lizcano [a], Aurora Pérez [b], Juan P. Valente [b]

[a] Open University of Madrid, UDIMA, Facultad de Enseñanzas Técnicas, Ctra, De la Coruña, km 38.500, Vía de Servicio, 15, 28400 Collado Villalba, Madrid, Spain
[b] Technical University of Madrid, School of Computer Science, Campus de Montegancedo, s/n, 28660 Boadilla del Monte, Madrid, Spain

ARTICLE INFO

Article history:
Received 24 January 2014
Accepted 5 June 2014
Available online 16 June 2014

Keywords:
Medical data mining
Time series analysis
Event
Classification
Electroencephalography
Stabilometry

ABSTRACT

There are now domains where information is recorded over a period of time, leading to sequences of data known as time series. In many domains, like medicine, time series analysis requires to focus on certain regions of interest, known as events, rather than analyzing the whole time series.

In this paper, we propose a framework for knowledge discovery in both one-dimensional and multidimensional time series containing events. We show how our approach can be used to classify medical time series by means of a process that identifies events in time series, generates time series reference models of representative events and compares two time series by analyzing the events they have in common.

We have applied our framework on time series generated in the areas of electroencephalography (EEG) and stabilometry. Framework performance was evaluated in terms of classification accuracy, and the results confirmed that the proposed schema has potential for classifying EEG and stabilometric signals.

The proposed framework is useful for discovering knowledge from medical time series containing events, such as stabilometric and electroencephalographic time series. These results would be equally applicable to other medical domains generating iconographic time series, such as, for example, electrocardiography (ECG).

1. Introduction

Knowledge discovery in databases (KDD) is a non-trivial process that aims to extract useful, implicit and previously unknown knowledge from large volumes of data. Data mining is a discipline that is part of the KDD process and is related to different fields of computing like artificial intelligence, databases or software engineering [1]. Data mining techniques can be applied to solve a wide range of problems.

There are many data mining techniques and algorithms for analyzing single-valued data. But more and more domains are generating large volumes of continuous data streams. Because of its particularities, this data type, called time series, requires special-purpose analysis techniques.

Formally, a time series can be defined as a sequence TS of time-ordered data $TS = \{TS_t, t = 1,\ldots,N\}$, where t represents time, N is the number of observations made during that time period and TS_t is the value measured at time instant t. In some domains, the value of not one but several observations might be measured at each time point, leading to multidimensional time series.

Time series analysis poses many problems, which are addressed by different data mining techniques. One key problem is to compare two time series, that is, to determine a measure of similarity indicating how alike the two time series are. The comparison of two time series is one of the major problems in time series analysis. If we have a measure of similarity between two time series, we can identify different patterns in a set of time series, search for one particular pattern in a time series, simplify a set of time series by finding similar time series, etc. In sports medicine, for example, it can be used to measure the progress of an injured athlete's recovery. Most existing techniques compare one whole series with another whole series [2,3]. However, there are many problems where it is requisite to focus on certain regions of interest, known as events, rather than analyzing the whole time series [4]. This applies to many different domains. An illustrative example outside the field of medicine is seismography, where the points of interest occur when the time series shows an earthquake, volcanic activity leading up to the earthquake or replicas.

* Corresponding author.
E-mail address: juanalfonso.lara@udima.es (J.A. Lara).

http://dx.doi.org/10.1016/j.jbi.2014.06.003
1532-0464/

From the viewpoint of information theory, the concept of time series event is closely related to the concept of entropy. System entropy is the amount of information that a set of system symbols contain. If systems are in this case time series, the events are the regions of the time series that contain most information, that is, that have greater entropy [5].

Time series that contain events are analyzed by comparing the events in one time series with the events in another to determine which event types the two time series have in common. To do this, there has to be a mechanism for calculating the distance between each pair of events in the two series.

The conception of what an event is varies from domain to domain. Suppose, for instance, that the events of the time series in a particular domain are the peaks generated by the local maxima. Given two time series, S_A and S_B (Fig. 1a), there are two regions of interest in series S_A and three in series S_B. Let us assume that in this particular domain the interesting features of the events are duration and amplitude (Fig. 1b). Comparing the two series, we find that the first event in S_A (E_{A1}) is very like the second in S_B (E_{B2}) because both events have a similar duration and amplitude. The third event in S_B (E_{B3}) is also very like the second in S_A (E_{A2}). In this case, series S_A and S_B have two events in common and are, therefore, very alike.

Another key data mining problem is the construction of a reference model from a set of time series. A reference model of a set of time series can be regarded as the time series that best represents that set. Experts can gain useful, clear and structured knowledge about a group of individuals from the reference model of a set of time series because the reference model contains the elements or patterns that the time series associated with those individuals have in common. Reference models built from time series of individuals suffering from a particular disease are very useful in medicine for confirming or ruling out whether other patients have the disease in question. Like time series comparison methods, most reference model generation techniques analyze the whole time series, and are not, therefore, applicable to time series that contain events.

The extraction of knowledge from time series that contain events is a challenging, open and unique problem that existing techniques do not cover as they analyze the whole time series.

To solve this problem, we propose a framework for event-based knowledge discovery. The main contributions of our framework described in this article are:

1. A method for comparing two time series that compares the events in the above time series.
2. A method for generating reference models in time series based on cluster analysis.

The preprocessing tool used in these contributions is a time series event definition language applicable to one-dimensional and multidimensional time series, which we proposed and validated in a previous research [5]. Domain experts can use this language to simply and straightforwardly define events likely to appear in domain time series. Other approaches regarding event identification are [6,7].

We believe that our framework advances the field of medical data mining, as it addresses the challenge open problem of knowledge extraction from medical time series that contain events. In fact, our framework has many potential uses in medical data mining, like clustering, outlier detection and classification of medical time series that contain events.

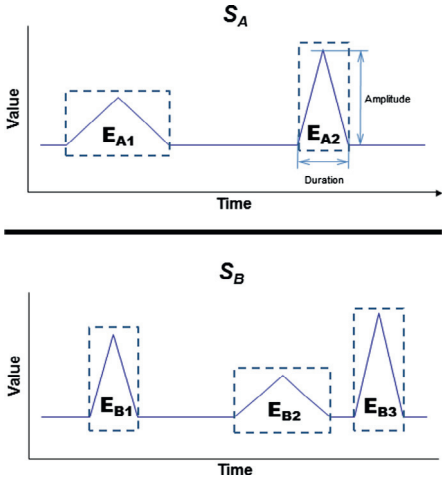
The principal aim of this article is to describe the above contributions and illustrate its use for classifying medical time series that contain events.

We believe that the proposed framework has several advantages that make a major contribution to the field:

- It is able to extract knowledge from medical time series that contain events. Users can address data mining problems in domains that generate this type of time series that existing techniques are unable to process.
- It is able to calculate a measure of similarity between medical time series. This solves higher level problems like, for example, clustering or outlier time series detection.
- It is able to build interpretable reference models, which is always a plus in medical data mining. This can help experts to gain a better understanding of their discipline and thereby favor its progress.

Our framework is a combination of contributions that can be used to address multiple problems regarding knowledge discovery in medical time series containing events. One such problem is the classification of time series, which is the focus of this article.

Generally speaking, the application of computer science to medicine is not a new development [38–48]. Data mining, for example, has very often been applied in the medical domain,



(a) Time Series S_A and S_B

Event ID	*Duration*	*Amplitude*
E_{A1}	5.1	2.3
E_{A2}	3	4.3
E_{B1}	2.8	3.9
E_{B2}	4.8	2.2
E_{B3}	2.9	4.2

(b) Features of Events

Fig. 1. Two time series and their event features.

especially to solve regression and classification problems. Some of the major data mining applications over recent years in different branches of medicine are summarized in [75]. This survey reveals that patient classification for diagnostic purposes is the most common data mining application. In this respect, different classification techniques (neural networks and decision trees) have been applied as a diagnosis support tool for heart disease [76], breast cancer [77], diabetes [78] and childhood obesity [79], etc.

However, most of these applications operate on data tables without a time component, unlike the research reported in this paper which focuses on the analysis of time series. Specifically, we have applied the proposed framework to time series generated in two medical domains: electroencephalography (EEG) and stabilometry. EEG is a neurological exploration used to diagnose nervous system disorders; stabilometry is a branch of medicine that investigates human balance. The positive results attest to the validity of the framework for addressing time series classification in the above domains. The framework is equally applicable to electrocardiography (ECG), magnetic resonance (MR) and computerized axial tomography (CAT), where it is useful for searching and locating neural correlates of mental activity.

The proposed framework (described in Section 3) will be applied to this type of time series. The results of applying the framework will be discussed in Sections 4 and 5. Section 6 will analyze the scope of application of our proposal, whereas the findings and future work will be detailed in Section 7. First, we will present work related to our research in Section 2.

Table 1 summarizes the notation used in the paper.

2. State of the art

Many time series data mining techniques are reported in the literature. As discussed in Sections 2.1 and 2.2, there are several proposals for comparing and extracting reference models from two time series. However, in domains where time series contain events, the proposals fall short, because, as far as we know, they use all time series regions for the comparison, even though some of them are potentially of no interest to experts. Section 2.3 describes some of the most interesting research on event identification. The proposed framework has been used to classify time series, a popular data mining problem that is described in Section 2.4.

Table 1
Symbol table.

Symbol	Explanation
TS	Any time series
N	Number of timestamps in a time series
TS_t	Timestamp of the TS at time t
S_A, S_B	Time series for comparison
E_{Ai}	ith event of time series S_A
E_{Bj}	jth event of time series S_B
D	Matrix of distances among events in S_A and S_B
D_{ij}	<i,j>th cell of matrix D
S	Set of time series
n	Number of time series in set S
S_i	ith time series in set S ($i = 1,\ldots,n$)
K	Number of problem classes
C_i	ith class of problem ($i = 1,\ldots,K$)
M_i	Reference model that represents time series in class C_i
M	Any reference model
S_{NEW}	Time series for classification
C_j	Class assigned to S_{NEW} ($j = 1,\ldots,K$)
q	Total number of events in the time series under analysis
E_p	pth event of all events ($p = 1,\ldots,q$)
m	Typical number of events in the time series under analysis
k	Number of clusters after event clustering
G_i	ith cluster of clusters after clustering ($i = 1,\ldots,k$)
r	Mean number of events per cluster
E_{Gi}	Most representative event of the event cluster G_i

2.1. Time series data comparison

Time series are used in many domains. Over the last few years, a lot of research related to time series has been carried out in domains as far apart as medicine [8], the financial sector [9] or traffic control [10].

There are many techniques for comparing time series and extracting common subsequences. A technique for comparing time series based on the Fourier transform is proposed in [1]. The aim is to extract a number of coefficients from the time series using the discrete Fourier transform, that is, by switching from the time to the frequency domain. Techniques based on alternatives to the Fourier transform, like the wavelet transform [3,11,12] have also been proposed.

The landmarks-based technique [13] proposes a method for comparing time series where the singular points (landmarks) of the curve, that is, the maximums, minimums and break points, have to be stored. It proposes the use of six transformations that are applied to the landmarks. Several features of the landmarks are invariant to the application of certain transformations, and only the invariant features of each transformation are taken into account when looking for series that are similar under certain transformations [14].

According to the clustering techniques taxonomy described in [67], the above proposals belong to the group of feature-based clustering techniques. These techniques bear some resemblance to our proposal which is also based on feature extraction, except that our proposal characterizes different events of interest rather than the whole time series.

Other proposals operate directly on the time series. Some of the most interesting approaches use the time warping distance concept. Time warping is based on the idea that two series are similar provided that the distance between the two series is less than a certain threshold when one time series is compressed on the time axis [15,16]. Another type of technique uses minimum bounding rectangles (MBR) to compare time series [17].

The last group of techniques is based on the use of models to compare two time series. Different and innovative proposals have emerged within this group. Some of these methods use Markovian models to compare time series [18], others use models based on graph theory [19], others again are based on comparing time series by looking at how they change shape [20], etc.

Of the above, the wavelet-based technique bears most resemblance to our proposal, as it is to some extent able to identify events. The drawback of this proposal is, however, that the identified events (wavelets) do not necessarily match the segments of interest to domain experiments. The other techniques discussed in this section are useful for comparing two whole time series. These techniques apply different methods to extract information on the entire time series. It is essential in many domains, like EEG or stabilometry, to focus only on regions of interest (events) in the time series. The framework proposed here aims to find a solution to the problem of comparing this type of time series that the above techniques do not cover.

2.2. Time series reference model

Apart from comparing time series, techniques designed to generate a representative model from a set of time series are of special interest for the research reported here.

There is a group of techniques that, before generating the model, propose a time series dimensionality reduction based on transforms, like the Fourier [2] or wavelet transforms [3]. In this case, the coefficients of the respective transform are obtained and then used to generate models by calculating their mean value, for example. In [21] a dimensionality reduction is also proposed as

a basis for iteratively building a model by searching the most specific generalization that clusters similar regions of the time series.

The above techniques build reference models from the values of all points in the time series. The proposal developed in our research, however, aims to analyze the periods of the series that are of special interest. In this respect, it bears some similarities to the technique presented in [22], which adopts the approach of first transforming the time series to a set of events. Chen et al. consider an event to be a subsequence of a time series that satisfies certain conditions set in each case [22]. They share our view of event introduced by Povinelli in [4]. They analyze the time series transformed to event sequences, using a technique called interval temporal logic. Even though it requires a previous step to transform the time series to the same magnitude, this course of action is highly versatile. The technique described in [22], which is very useful for discovering the temporal relations between series events, requires major low-level changes to define the conditions determining which regions of the time series are of interest. On the other hand, our proposal attempts to solve the problem of the high specificity of existing solutions by setting up a general-purpose framework that is easily adaptable to multiple domains.

Unlike the above, there are model generation methods that deal directly with the time series without any previous dimensionality reduction or simplification step. This applies to the work presented in [23]. This proposal is one of a group of techniques with a philosophy based on extracting time series segments (patterns) that are common to a significant number of time series and could therefore be a feature of a group. It defines a pattern extraction algorithm with no pattern length specification. Other noteworthy papers likewise addressing the problem of detecting subsequences which frequently appear in more than one time series are [24–26].

In some domains, the above techniques can be troublesome as it can generate reference models with segments that are of little or no interest to domain experts. Our framework reduces dimensionality by locating and characterizing events using a set of interesting attributes. Domain expert knowledge is added for this purpose. This prevents the above problem, which is the result of some time series regions being meaningless.

Another possibility is to apply some of the clustering techniques reported in [67] to build reference models. We have applied clustering techniques in our research in order to classify a time series by determining its similarity to different reference models generated as archetypes of each class rather than for modelling purposes.

2.3. Events

A major issue regarding time series where the interesting information is confined to certain regions of interest is how to identify those regions or events. Event identification can be considered first and foremost as a preprocessing step in preparation for later data mining tasks.

In this respect, the field of temporal abstraction is widely recognized as a consolidated intermediate step in understanding and processing time series, as illustrated in [68,69]. The acquisition and maintenance of domain-specific temporal-abstraction knowledge used to create meaningful interval-based abstractions from raw time-stamped clinical data is described in [68]. In [69], previous research into the development of intelligent clinical data analysis systems that incorporate temporal abstraction mechanisms is surveyed. However, most of these papers make no mention of the concept of time series event as the key component for driving temporal abstraction.

Other earlier research is based on the search for events within time series. Research reported in [70] develops a unification-based temporal grammar designed to define time series sequences and events. A sequence is regarded as a time interval in which the measured value of a particular single-valued attribute remains close to a specified threshold value, whereas an event is an interval in which two or more sequences are superposed. Domain experts unfamiliar with this complex events notation based on single-valued sequences would find the grammar hard to use unless they were expert programmers. However, this research lays the foundation for the ideas proposed here: temporal abstractions of time series as events, one of the most successful trends in the field of KDD for multidimensional and real-world clinical data, as well as null hypothesis testing, as reported in [71]. This type of event-based abstraction was developed for TRACE [72], a graphical time series analysis and annotation tool for selecting regions of interest, which is useful for reducing database complexity and selecting time intervals of relevance in a particular domain, although it cannot explicitly annotate events within the series.

There are other papers dealing with the topic of events identification in time series. In [27], they propose a way of defining events using a function of interest that evaluates each and every subsequence in the time series with the resulting computational cost. Apart from this, there are not, as far as we know, many proposals focusing on time series events definition. What we more often come across are methods for dividing a time series into subsequences. These methods are based, for example, on statistical concepts like the change point [28] or the use of genetic algorithms [29]. Additionally, the proposed techniques tend to be domain specific and are not easily applicable to other domains as the source code of the applications would have to be modified at a very low level for this purpose. This applies to the TSDM framework [30], proposing a set of methods for identifying events in the financial domain. Likewise, most event processing methods are applicable to one-dimensional time series. This way, they obviate the complexity of multidimensional time series, where there are possible dependencies among the different time series attributes (dimensions). A specialized method for identifying T-patterns in the field of psychology is proposed in [31]. This method is very closely coupled to the domain for which it was devised and hard to use in other areas.

The framework developed as a result of our research aims to overcome the weaknesses of the above approaches with respect to portability and multidimensional time series issues.

Regarding event management, there is another important line of works, which is related to causality detection. In other words, two time series can be very similar except that one happens before another. This line of research, which differs from the aim of our research, has already been dealt with at length in the literature [32,33].

For an exhaustive description of the language elements and rules that govern events definition, see [5].

2.4. Time series classification

Time series classification has gained in importance in recent years. The most precise and robust proposals are based on the simple nearest neighbor algorithm [34–36]. Despite its accuracy and robustness, the nearest neighbor algorithm has sizeable drawbacks like, for example, its high computational cost or the fact that it does not indicate why an object is placed in a particular class. Examining this algorithm from the viewpoint of the event problem with which we are concerned, the nearest neighbor algorithm has another major weakness: it does not enact an introspective time series event analysis process. For some reason, the analysis of the distance between the events in the time series takes precedence over the analysis of the distance between time series when the time series contains events.

In this respect, an approach more like our proposal is reported in [37]. It is based on locating distinctive subsequences in time series. These sequences do not necessarily have to match the segments of interest to experts. The representative of each class has to be a continuous segment of the time series, which, from the viewpoint of the problem of classifying times series with events, is again a disadvantage. Generally, the reference model of a class will not necessarily be a single event; it may be defined by the expert as a combination of multiple events that take place at different times.

Note, with regard to time series dimensionality, that multidimensional time series are harder to classify. In this respect, other authors have explored the multidimensional time series classification problem. The research described in [73], developing a temporal abstraction framework for generating multidimensional time series features suitable for classification tasks, is a prominent example. It proposes an algorithm called STF-Mine that automatically mines discriminative temporal abstraction patterns from the time series data and uses them to learn a classification model.

Multidimensional time series classification has also been addressed more recently in [74], proposing a framework for multidimensional time series classification that weights the class prediction from each time series dimension. These weights are based not only on each stream's previous track record on the class it is currently predicting, but also on the distance from the unlabeled object. Albeit similar to our idea, neither this nor the previous proposal distinguishes the regions of interest to experts and are thus of limited applicability for solving the problems stated in this article.

3. Proposed solution: a framework for classifying time series containing events

This section describes the proposed framework for classifying time series containing events.

3.1. Time series classification strategy

In Section 2, we introduced some of the problems that data mining techniques can deal with. We also mentioned several proposals that have emerged lately to solve such problems. Despite the efforts in the data mining field, however, many problems remain open.

The main disadvantage of the techniques reported in the literature is that they analyze whole time series, obviating the fact that the only worthwhile analysis in many domains is of regions of interest.

In this paper we propose a general framework for knowledge discovery in time series applicable to domains, like medicine, where the time series contain events of interest for experts. In particular, this article shows how the proposed framework can be used to classify one-dimensional and multidimensional time series. Generally, the steps to be taken to apply this framework to any particular domain are as follows:

1. Understand domain data.
2. Define domain events. The different types of events that can occur in time series have to be defined. To do this, first, it is necessary to seek expert help to establish the conditions determining which time series regions are events. Based on these conditions, the different types of events will be defined using the language proposed by the authors in [5].
3. Implement an ad hoc method in a high-level language to extract the key features of the domain events.

As we have seen, traditional techniques have weaknesses as regards domains with event-based time series, such as failure to consider events, domain dependency (very low-level changes to the application source code are required for use in different domains) or non-portability. Because of these weaknesses, experts find such techniques very hard (and impossible in some cases) to apply, leading to a waste of vast quantities of potentially useful data. This framework is a tool specially designed to help domain experts extract knowledge from time series and overcome the deficiencies of other techniques. Our proposal is a special-purpose tool for knowledge discovery in time series based on event analysis.

The reported framework can be considered as a classifier that is able to classify a new time series (S_{NEW}) according to a predefined set of classes C_i ($is = 1,2,\ldots,K$). To do this, we have developed two methods that constitute the groundwork of our framework:

1. A method for comparing two time series that contain events.
2. A method of generating reference models based on a set of time series containing events.

Apart from the above two methods, we have also proposed a time series events definition language. This events definition language enables domain experts to simply and naturally define which events are likely to appear in the domain time series. The time series classification process uses the time series events identified by this mechanism as part of a strategy that combines the use of a method for comparing two time series and a method for generated reference models based on a set of time series that contain events. The main feature of these methods is that they are based on a detailed analysis of the time series events. They are therefore applicable in domains not covered by the proposals reported in the literature. The time series classification strategy is as follows:

1. Generate, for each class C_i ($i = 1,2,\ldots,K$), a reference model (M_i) from a set $S = S_1,S_2,\ldots,S_n$ of training time series. The set of time series should be large enough for each considered class. In statistics, a sample of size $n > 30$ is considered to be representative enough.
2. Compare the new time series (S_{NEW}) with each reference model M_i ($i = 1,2,\ldots,K$) created in step 1.
3. Select that class C_j whose reference model M_j is most similar to the new time series S_{NEW}, such that $C_j = C_i \mid \text{similarity}(S_{NEW},M_j) = \min(\text{similarity}(S_{NEW},M_i))\forall i = 1,2,\ldots,K$.

This strategy is illustrated in Fig. 2, which offers an overview of the proposed framework structure. The framework receives a new time series, S_{NEW} **(a)**, and returns as a result the particular class, C_j **(b)**. To do this, the classifier module has to read the previously generated reference models (one for each class) in a database and compare each model against the new time series **(c)**. The selected class will depend on the resulting similarity. To do this, the framework includes a method for comparing time series.

One reference model for each class $i = 1,2,\ldots,K$. M_i has to have been previously generated to be able to classify the S_{NEW} series. The module responsible for generating these reference models based on a set of time series S is the reference model creator module **(d)**.

Clearly, both the time series comparison method and the reference model generator method require knowledge of events present in the time series under analysis. In order to compare two times series, we first have to extract the events that they contain. Additionally, time series events also have to be extracted beforehand in order to generate a reference model of a set of times series. Therefore, the event definition language is a tool providing input for the other methods. Within the framework, the event definition

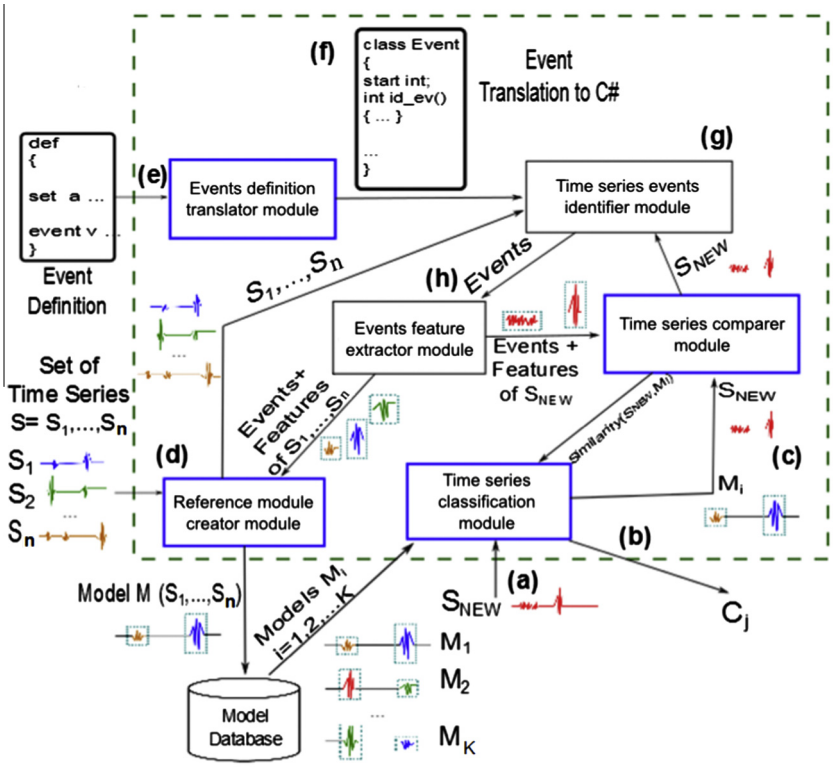


Fig. 2. Framework for data mining time series containing events.

language can be viewed as a data preprocessing module thanks to which the time series dimensionality is reduced by extracting events that are processed by the other framework methods.

To do this, there is a module that identifies and outputs a list of events occurring in the input time series. The events identifier method is generated automatically from a particular events definition **(e)** by the domain expert using the events definition language proposed in this framework. This events definition is translated by a translator module that generates source code implementing an events identifier method in a high-level language, C# in this case **(f)**. This source code is the input and kernel of the events identifier module **(g)**.

The identified events are analyzed by the event feature or attribute extractor module **(h)**. The events feature extractor module is the only specialized part of the framework and has to be implemented ad hoc for each domain. A typical feature extracted will be the timestamp at which the events take place. This feature can be used by our framework to analyze the order relationship between events contained in time series.

In order to classify times series, our framework uses the functionalities for comparing time series and building reference models, and the proposed mechanism for identifying events. This mechanism operates as a preprocessing tool in our framework. These three contributions are described in the following.

3.2. Events definition language

The proposed framework for classifying time series requires a mechanism for identifying events. There are few proposals in the literature that deal with the problem of event identification [27–31]. In actual fact, methods usually divide a time series into subsequences based mainly on statistical concepts such as the change point. Many such techniques are not general enough for use in more than one domain, as no two domains are the same, and therefore the conditions determining whether or not something is an event are particular to each domain. Indeed, very low-level changes to the source code are required in order to apply these techniques in other domains.

This is compounded by the circumstance that most techniques deal with events in one-dimensional time series, obviating the complexity arising when we have multidimensional time series, as there may be dependencies among the different attributes (dimensions) of the time series.

To overcome the weaknesses of existing techniques, which may be too specific for use in different domains and with multidimensional time series, we have proposed a formal events definition language enabling domain users to define events in domain time series. The language is of no use for defining events without the participation of domain experts. However, events are easy and intuitive for experts to define using a very high level language, which, as we will see in the examples, is very like natural language.

Apart from the events definition language, we have developed a translator that checks whether an events definition obeys the language rules and is lexically, syntactically and semantically correct. Otherwise, errors are reported to the user. If the events definition is correct, it is translated to source code in a high-level language (C# in this case). In this section, we give an overview of the proposed language and the developed translation tool.

To specify the events definition language, we have enacted a process that aims to simulate how human beings visualizing a time series naturally locate events. To identify events, people intuitively try to locate points in the series that have certain particularities. Generally, these are points at which the series takes an outlier value with respect to the other points of the series. These points are usually maximums, minimums, points at which the series intersects a particular value, etc. Sometimes, these points of interest have to be filtered to determine which are really interesting and which do not meet the necessary conditions to be so. After locating the points of interest, human beings search backwards in the series for the point that meets a particular condition signifying the start of the event. Similarly, they search forward in the series to determine where the event ends. Normally these start and

end points are points at which the series is or returns to normal before and after the occurrence of the event.

The proposed language uses basic concepts of set theory, logic, algebra and descriptive statistics. Set theory is necessary as the points of interest of the series are located by means of a process of refining more general sets of points as described earlier. Logic is required to establish the conditions that determine what a point of interest in the series really is and what conditions such points have to meet to be part of an event. Algebra plays a fundamental role as arithmetic operations are needed to perform calculations on sets of points of interest in the series and their respective value. Finally, descriptive statistics is used to represent the normality or stability of the series, where such normality is represented by means of descriptive measures of centralization and dispersion, such as the mean, mode or standard deviation. These measures are used to determine regions of the series that are removed from normality and have high entropy and which, therefore, are potential events.

From the formal viewpoint, the proposed language uses an ALFEUC-type description logic [49], that is, a description logic that models concepts, roles and individuals and also includes functional properties to characterize events using descriptive statistics, full existential qualification to algebraically model the comparison against models or patterns and concept union and negation to process time series according to set theory. Some of the elements of the language are predefined and others will have to be defined for each particular domain. The aim is to offer a simple language that is comprehensive enough to be generally applicable.

The events definition procedure is as follows:

1. Define the set of points of interest in the time series, which will be the basis for defining events. Note that these sets refer to **time instants** (timestamps) and not to the value that the time series takes at that time instant. Therefore, these sets will be ordered, will not contain repeated elements and will include only positive integer values.
2. Use of the basic language elements and sets of points of interest to define events.

We have devised a series of basic predefined elements for the events definition language that will usable in any domain as an aid for users who will not have to define such elements. These elements are outlined in Table 2.

The language also includes other basic elements present in any language, such as, for example, identifiers, numerical and logical constants.

The body of each events definition is delimited by braces and preceded by the reserved word *def*. The *def* body contains the three types of elements described below. An events definition can include any number of elements provided that they are arranged in the following order.

i. Statement of time series
The time series can be either multidimensional or one-dimensional, but only real number series are considered (if they were integer value series, all the values would be transformed to real numbers). To define a time series, we use the reserved word *series* and then we specify the name of the time series. A statement can include any number of time series (at least one), where each statement is separated by a semicolon (;). The first two rows of Table 3 specify the details of the syntax for stating time series together with several examples.
No two time series can have the same name. Once a time series has been defined, it will then be able to be used for defining statistics and basic sets of time series. Additionally, time series will also be able to be used as part of an arithmetical expression.

ii. Definition of sets of points of interest
The sets of points of interest contain timestamps that the domain expert considers to be relevant and are the basis for later defining events.
A set of points of interest is always defined from sets previously defined or predefined in the language. This simplifies the expert's job. There are two options that we will detail below: the first uses conditions to define a set from elements of another pre-existing set; the second uses operators to define sets.
No two sets can have the same name. Once a set has been defined, it can then be used to define other sets and to define events. A definition can include any number of sets (even none), where each definition is separated by a semicolon (;).

- **Condition-based definition.** In this case, sets can be defined based on the elements of pre-existing sets by filtering with a condition. To do this, we use the notation described in Table 3. The *set_name'* identifier must correspond to a previously defined (predefined or derived) set. The language includes the following predefined sets: *max*, which contains all the timestamps of a time series in which there exists a local maximum, and *min*, which contains all the timestamps of a time series in which there is a local minimum. The scope of the id_1 identifier is confined to the braces that are used to define the set. Table 3 includes an example of a set called *MaxMultiples50* composed of the all the timestamps of the time series TS_A where there is a local maximum and that are multiples of 50.
- **Operator-based definition.** In this case, a set is defined from another two sets defined previously using the union, intersection and difference operations. Again the notation used is listed in Table 3. In this case, *new_set* is the name of the new set that is to be defined and old_set_1 and old_set_2 are previously defined sets. Neither old_set_1 nor old_set_2 can bear the same name as *new_set*. On the other hand, OPERATOR can take the values of *joint* (union), *intersec* (intersection) or *diff* (difference).
Table 3 includes several examples of the definition of sets using the above operators.

iii. Definition of events proper
An event type is always defined using points of previously defined sets, and an event peculiar point, plus the event start and end points, are determined. The events definition notation is listed in the bottom two rows of Table 3. The *set_name*, set_name_1 and set_name_2 identifiers must correspond to the previously defined sets. These can be the same or different sets. The scope of the *peculiar_point* is confined to the braces used to define the events.

The bottom row in Table 3 includes an events definition example that uses the set *MaxMultiples50* defined earlier in Table 3. Suppose that, in a particular domain, the points of this set are the peculiar points and that the events start at a timestamp before and end at a timestamp after these peculiar points. Considering that *previous* and *next* are two predefined language operators (respectively returning the previous and next element to one given in a particular set), this type of events are defined as illustrated in the bottom row of Table 3.

The conditions used to define sets of points and events can be built from expressions and relational and set operators. Additionally, compound conditions can be defined using the logic operators

Table 2
Basic events definition language elements.

Basic elements	Description	
Time series	***Definition***	Ordered set of measurements in a time interval; dimensions are conceived as interdependent mathematical functions in multidimensional time series
	Example	stockexchange_price **series;**
Statistical measures	***Definition***	A series of basic predefined statistical measures is defined for each dimension
	Examples	Average (**avg**), mode (**mode**), median (**med**), standard deviation (**stdev**) and variance (**var**)
Sets of predefined points	***Definition***	Some points tend to be of interest in most domains, like, for example, the moments in the series where a maximum is reached
	Examples	Series timestamps (**timestamps**), local maxima (**max**) and local minima(**min**)
Arithmetical operators	***Definition***	Basic arithmetical operators
	Examples	**+, −, *, /,** etc.
Relational operators	***Definition***	Operators that are useful for comparing and establishing relationships among elements
	Examples	**<, >, <=,** etc.
Arithmetical functions	***Definition***	More sophisticated arithmetical functions
	Examples	Square root (**sqrt**), exponent (**exp**), absolute value (**abs**), etc.
Logic operators	***Definition***	Operators that can be applied to Boolean type data
	Examples	**AND, OR, NOT**
Set operators	***Definition***	Operators for handling sets
	Examples	$\varnothing, \cap, \bigcup, \supset, \supseteq, \subset, \subseteq, \forall, \exists$, etc.

Table 3
Events definition language syntax.

Events definition parts		Description	
Time series		***Syntax***	**series** series_name;
		Examples	**series** $series_1$;
			series $series_2$;
Sets of points of interest	Using conditions	***Syntax***	**set** set_name { id_1 **in** set_name' **such that** CONDITION};
		Example	**set** MaxMultiples50 **{**x **in max(**TS_A**) such that ((**x**%** 50**) ==** 0**)};**
	Using operators	***Syntax***	**set** new_set**=** old_set_1 **OPERATOR** old_set_2;
		Examples	**set** mySet**=** $mySet_1$ **joint** $mySet_2$;
			set mySet'**=** $mySet_1$ **diff** $mySet_2$;
			set mySet''**=** $mySet_1$ **intersec** $mySet_2$;
Events		***Syntax***	**event** event_name **{peculiar_point in** set_name, **start in** set_name_1, **end in** set_name_2 **such that** CONDITION **}**
		Example	**event** ev {**peculiar_point in** MaxMultiples50, **start in** TS_A, **end in** TS_A **such that** **previous(peculiar_point,** TS_A**) == start** **&& next(peculiar_point,** TS_A**) == end};**

AND ('&&') OR ('||') and negation ('!'). Again, expressions can be simple or compound. Simple expressions can be integers, real or Boolean. Compound expressions are defined using the different arithmetic operators and set operators.

Table 3 shows examples of one-dimensional time series. Note, however, that, thanks to its flexibility, the proposed language is able to create expressions involving the different dimensions of a multidimensional time series. This enables effective preprocessing of multidimensional series in search of events that are characterized for later use in data mining tasks, like, for example, classification.

For an exhaustive description of the language elements and rules that govern events definition, see [5].

3.3. Comparison method

The time series comparison method was designed to emulate the procedure that experts would enact. We examined a number of different medical domains, including the domains described in this article.

We had access in each domain to a panel of from three to five physicians, specialized in the respective field and without specialized data mining knowledge. Each panel was given different cases and asked to use conventional methods to solve the problem of comparing two time series using a process of iterative negotiation called the Delphi method [50,51]. During this process, the experts were observed and then interviewed to gather the knowledge required to automatically replicate this process. Thus, we elicited a series of guidelines and common criteria that experts use to visually compare two time series with events.

On this ground, our method is to some extent inspired, at least partly, by the experts.

We have observed that experts faced with the problem of comparing two times series containing events proceed as follows:

1. They take an event in the first time series and focus on its key attributes.
2. They then look for a similar event in the second time series. If they find a similar event, they pair off these events.

3. They take the next event in the first time series and repeat steps 1 and 2.
4. Finally, depending on how many events they have managed to pair off, they specify the degree of similarity between the series. This degree of similarity will, of course, increase with the number of common events that they have identified in the time series.

This process, common to all analyzed domain experts, is the groundwork of our method. Rather than establishing heuristics, the adopted design decisions have been taken in order to closely emulate the process described above, and have been modeled very generically in order to assure the applicability of our method in any domain in which time series contain events.

Formally, the aim of the comparison method proposed here is to find a function F that takes two time series S_A and S_B and returns a similarity value in the interval $[0,1]$, where 0 indicates that the two series are completely different and 1 denotes that the two series are identical, as described in (1).

$$\begin{aligned} &F : TS, TS \rightarrow [0, 1] \\ &F : S_A, S_B \rightarrow Similarity(S_A, S_B) \end{aligned} \tag{1}$$

To determine similarity, the proposed method looks for events that appear in both series. The more events that the two series under comparison have in common, the closer similarity will be to 1. The similarity of series that do not have any event in common will be equal to 0.

To determine whether an event in one time series appears in another, the event has to be characterized by means of a set of attributes (which are normalized to prevent any attributes dominating others) and compared with the other events in the other series. To speed up this process, all the events present in the two time series are clustered. Therefore, if two events belong to the same cluster, they are similar. The goal is to find events that are members of the same cluster and belong to different time series.

Therefore, the proposed algorithm for extracting events common to two time series S_A and S_B is:

Algorithm 1. Time series comparison

Input: Two time series, S_A and S_B.
Output: Similarity (S_A,S_B).
cluster all events E_m of both time series //*events that appear in S_A or in S_B*
for each cluster extracted from previous step **do**
  while there are events of S_A and S_B in the cluster **do**
    create all possible event pairs (E_{Ai},E_{Bj}), where $E_{Ai} \in S_A$ and $E_{Bj} \in S_B$;
    select the event pair that minimizes distance (E_{Ai},E_{Bj});
    //Manhattan distance is used. This extracts the two most alike events that are
    //common to both series because they are in the same cluster, belong to
    //different time series and are nearest to each other
    delete events E_{Ai} and E_{Bj} from the cluster;
    return the pair (E_{Ai},E_{Bj}); *//(E_{Ai},E_{Bj}) is an event common to both series*
  end
end
return the similarity between S_A and S_B; *//similarity is calculated according to (2)*

The algorithm extracts events common to the two series, yielding a set of pairs of events such that pair (E_{Ai},E_{Bj}) indicates that event i of series S_A is equal to (or similar enough to be considered as equivalent) to event j of series S_B. These pairs of common events are used to determine the similarity value between time series using (2) in which E_p denotes each of the events identified in both series (q is the total number of events considering S_A and S_B).

$$SIM(S_A, S_B) = \frac{\sum_{i,j} length\_pair((E_{Ai}, E_{Bj}))}{\sum_{p=1}^{q} length(E_p)} \tag{2}$$

In (2), *length_pair* is the length of pair (E_{Ai},E_{Bj}) that is determined by (3).

$$length\_pair((E_{Ai}, E_{Bj})) = length(E_{Ai}) + length(E_{Bj}) \tag{3}$$

On the other hand, *length* is the duration of a particular event E_p, which, as shown in (4), is the difference in absolute value of the time at which the event ends less the time at which it starts.

$$length(E_p) = |final\_timestamp(E_p) - initial\_timestamp(E_p)| \tag{4}$$

The idea behind Eq. (2) is to compare the amount of a time series that is common to the two time series (numerator) with respect to the total amount of the useful time series, that is, with respect to the total length of the events (denominator). The numerator and, therefore also the similarity value, will increase in proportion to the number of common events. If there is no event in common, the numerator and the similarity value will be 0. If all the events that are extracted are common, the similarity value will be 1. Exceptionally the value of the denominator could be 0 if there are no events to analyze. In this case, the similarity value will be 1.

The reasoning underlying Eq. (2), which is useful for calculating the similarity between time series, is really the same as the Jaccard similarity coefficient. This coefficient, whose formula is shown in (5), measures the similarity between two sets of elements A and B, defined as the intersection between the two sets (elements in common, $A \cap B$) divided by the union of the two sets (all the elements in both sets, $A \bigcup B$) [52].

$$SIM\_Jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{5}$$

We studied other measures of similarity, such as Sorensen's similarity index [53], but we opted for the use of a similarity formula inspired by the Jaccard index, as it is especially suited for quantitative data, is a normalized similarity measure and takes up the idea of identifying common elements inspiring our method of comparison.

Some aspects of the event clustering process require further explanation. To cluster events, it is necessary to calculate the distance between each pair of events. The distance between events is also used when identifying pairs of similar events. In both cases we opted to use the *Manhattan* distance [54]. This distance calculates the sum of the absolute differences between each of the event coordinates. Note that the event coordinates are the attributes of interest that characterize the events (for example, event amplitude and duration in Fig. 1). These events will typically include the timestamp identifying when the event occurs within the time series, as this will provide for a richer comparison of events by including information on the total order of the occurrence of events in the time series. These attributes will have been calculated in a previous dimensionality reduction step as explained in Section 3.1 (see Fig. 2h). We looked at other distance measures, but finally opted for the *Manhattan* distance because the mean distance per attribute is used during the clustering process to determine whether two elements are members of the same cluster. This mean distance per attribute is obtained straightforwardly by dividing the total distance by the number of attributes. The use of the *Manhattan* distance then saves time as it obviates additional transformations that would make the clustering process more complex to develop and more computationally intensive.

We have used hierarchical clustering techniques based on the generation of ordered cluster sequences (hierarchies). The hierarchical structure is represented as a tree called dendrogram.

There are several cluster similarity criteria for combining the two most similar clusters in an interaction. In this research, we have used average-link, according to which the distance between two clusters is calculated as the mean of the distances between the elements of both clusters.

The implemented algorithm follows the principles applied in AGNES (*AGglomerative NESting*) [55], one of the best-known bottom-up hierarchical clustering algorithms. Hierarchical algorithms rank among the most popular data mining algorithms, because, among other things, they are fast to compute. Additionally, taking into account that the proposal described here should be a general-purpose method and there is no a priori information for specifying the optimum number of clusters in each domain, bottom-up hierarchical clustering is a good option, as there is no need to specify the number of clusters k beforehand. After building the dendrogram, a parameter has to be set to determine the cut-off level for cluster formation within the dendrogram. In this respect, we decided to determine this parameter automatically as indicated in [56].

In order to compare two time series, it is necessary to extract their events. To do this, we have to analyze the two whole series, meaning that the complexity of event extraction is $O(2{\cdot}N)$, where N is the number of timestamps in each series. Then a hierarchical clustering algorithm is executed on the extracted events. This leads to a complexity $O(q^2)$, where q is the number of events in the two series. The clusters are analyzed in order to obtain the common events. To do this, we have to locate, for each event, the event with the shortest distance to that event. This process has a complexity of $O(q^3)$, which matches the overall complexity of the method of comparison, if we obviate the event identification process. Cubic complexity with respect to the number of events is in our opinion an acceptable complexity as $q \ll N$.

To illustrate how the time series comparison method works, we will use the example shown in Fig. 1. There are two time series, S_A and S_B, in the example. S_A contains two events (E_{A1}, E_{A2}) and S_B includes three events (E_{B1}, E_{B2}, E_{B3}). The first step of the algorithm enacts a clustering process on the above events. To do this, it first has to calculate the distance between each pair of events using the *Manhattan* distance. The result is the symmetric distance matrix D shown in Table 4. Each cell D_{ij} in this matrix represents the distance between events i and j. For example, cell $D_{21} = 2.05$ represents the distance between the event that occupies position 2 in the matrix rows (E_{A2}) and the event that occupies position 1 in the matrix columns (E_{A1}). The amplitude of event E_{A2} is 4.3 and its duration is 3. On the other hand, the amplitude of event E_{A1} is 2.3 and its duration is 5.1. These values yield a similarity value of $D_{21} = |4.3 - 3| + |2.3 - 5.1| = 1.3 + 2.8 = 4.1$. The other values of the matrix are calculated similarly using the *Manhattan* distance.

After calculating the matrix of distances among events, the events are clustered. The result for the example case is shown in Fig. 3, where there are two clusters, $G_1 = \{E_{B1}, E_{A2}, E_{B3}\}$ and $G_2 = \{E_{B2}, E_{A1}\}$.

The algorithm analyses each of the resulting event clusters to obtain the events that are common to both time series, as shown

Table 4
Matrix D of distances among events for the example in Fig. 1.

Distance	E_{A1}	E_{A2}	E_{B1}	E_{B2}	E_{B3}
E_{A1}	0	4.1	3.9	0.4	4.1
E_{A2}	4.1	0	0.6	3.9	0.2
E_{B1}	3.9	0.6	0	3.7	0.4
E_{B2}	0.4	3.9	3.7	0	3.9
E_{B3}	4.1	0.2	0.4	3.9	0

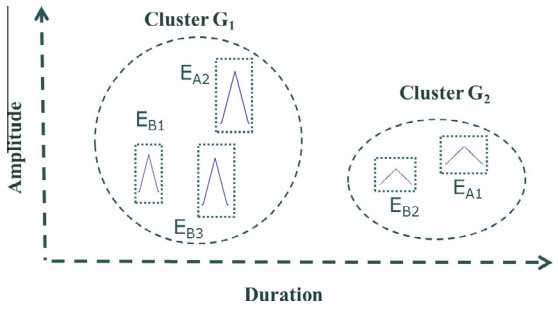


Fig. 3. Event clusters obtained for the example in Fig. 1.

Table 5
Matrix D of distances among events for the example in Fig. 1.

Iteration	Analyzed cluster	Are there events from both series in cluster?	Generated pair	Deleted events	Resulting cluster
1	$G_1 = \{E_{B1}, E_{A2}, E_{B3}\}$	Yes	$<E_{A2}, E_{B3}>$	E_{A2}, E_{B3}	$G_1 = \{E_{B1}\}$
2	$G_1 = \{E_{B1}\}$	No	–	–	–
3	$G_2 = \{E_{B2}, E_{A1}\}$	Yes	$<E_{B2}, E_{A1}>$	E_{B2}, E_{A1}	$G_2 = \{\varnothing\}$
4	$G_2 = \{\varnothing\}$	No	–	–	

in Table 5. The first iteration of the *for* loop analyses G_1. This cluster is also analyzed to check whether it meets the internal *while* loop condition. This condition is met in iteration 1, as cluster G_1 contains events from both series. This being so, we create all the possible pairs of events of the cluster $<E_{Ai}, E_{Bj}>$, where $E_{Ai} \in S_A$ and $E_{Bj} \in S_B$. These pairs are $<E_{B1}, E_{A2}>$, $<E_{A2}, E_{B3}>$, $<E_{B1}, E_{B3}>$. For each of those pairs, the distance between the two events of the pair is 0.6, 0.2 and 0.4, respectively. Therefore, the pair with a minimum distance between events is composed of events $<E_{A2}, E_{B3}>$. This pair is returned as the common pair of events and will be taken into account for the final similarity calculation. At the same time, the events that are members of the pair, E_{A2} and E_{B3}, are removed from the cluster and are not considered in future iterations. The resulting cluster $G_1 = \{E_{B1}\}$ is analyzed again. This time, the internal *while* loop condition is not met (iteration 2) and we move onto analyze the next cluster G_2. The analysis of this cluster is similar, and the iteration generates the pair $<E_{B2}, E_{A1}>$. The iterative process ends at iteration 4, where G_2 is empty and there are no more clusters to analyze.

At the end of this iterative process, we obtain the pairs $<E_{A2}, E_{B3}>$ and $<E_{B2}, E_{A1}>$ as common pairs of events. We use the above pairs to calculate the final similarity between the time series using the similarity formula proposed in Eq. (2). The result of the similarity calculation is 0.849, as shown in the following:

$$SIM(S_A, S_B) = \frac{length\_pair((E_{A2}, E_{B3})) + length\_pair((E_{B2}, E_{A1}))}{length(E_{A1}) + length(E_{A2}) + length(E_{B1}) + length(E_{B2}) + length(E_{B3})} = \frac{(3 + 2.9) + (4.8 + 5.1)}{5.1 + 3 + 2.8 + 4.8 + 2.9} = 0.849$$

The length of the events in time series similarity calculation Eqs. (2)–(4) is defined as the number of *timestamps* between the start and end of the time series. For clarity's sake, the computation of above example was simplified and the readily available event *duration* attribute was used instead of timestamps. The result is, in any case, equivalent.

3.4. Model generation method

Like the time series comparison method, the reference model generation method is designed based on the analysis of how experts enact this process. In this case, the same panels were used

as in the comparison method. Each panel was given different cases and asked to use traditional methods to solve the problem of reference model generation negotiating their decision over several rounds according to the Delphi method. Again, the experts were observed during this process and then interviewed to elicit the knowledge that was to be adopted in our framework.

The process elicited from experts facing the problem of building a reference model of a set of times series with events is as follows:

1. They take an event from the first time series and they fix its most relevant attributes.
2. They then search the other time series for similar events. They identify any events that they find as representative of the group and label them as candidates for membership of the final reference model.
3. They take the next event in the first time series and repeat steps 1 and 2. When they finish with the first time series, they move on to the other time series.
4. Finally, they analyze the candidate events and select the most representative, that is, events that appear in more time series, for inclusion in the model.

This method can, in a sense, be said to be prescribed by the experts, as it aims to emulate and, partly, automate how they intuitively perform this process.

The proposed model generation method receives a set of time series $S = \{S_1, S_2, \ldots, S_n\}$, each containing a particular number of events, and generates a reference model M that represents this set of time series. The model M is built on the basis of the most characteristic events. The most characteristic events of S are the events that appear in the highest number of time series of S.

To find out whether a particular event in a time series S_i also appears in another time series S_j ($j \neq i$), the event has to be characterized with an attribute vector (these attributes are normalized to prevent any attributes dominating others) and compared with the other events of the other series. To speed up this process, all the events present in the time series are clustered, so *similar* events belong to the same cluster (two events are similar if the value of the distance between them is less than a certain threshold). On the one hand, the clustering process is useful for identifying the different groups of events. On the other hand, it facilitates the extraction of the most characteristic events. The aim is to find the clusters in the set that contain events that appear in the highest number of time series, that is, characteristic events. Clusters of similar events are analyzed exhaustively in order to extract the event representative of each cluster. This will be described later.

These extracted representative events are the characteristic events of S and will be part of the final model. Let $S = \{S_1, S_2, \ldots, S_n\}$ be a set of n time series and m the typical number of events that appear in the time series of S. The algorithm for generating a reference model M representing the set S is as detailed below (for the purpose of making the algorithm more legible key decisions are justified at the end of the algorithm):

Algorithm 2. Time series reference model

Input: $S = \{S_1, S_2, \ldots, S_n\}$, where S is a set of n time series and m the typical number of events that appear in the time series of S.
Output: M, reference model of S.
M = Ø; //*initialize the model*
Determine the typical number of events m;
//*m is the typical number of events in each time series of S. We will discuss how*
//*to determine this value at the end of the algorithm*
cluster events; //*events that appear in time series from S*
while $m > 0$ **do**
  get the most significant cluster G_i;
  //*cluster significance is measured using (6)*
  extract the event E_{Gi} that best represents the cluster G_i;
  //*extract the event that is most representative of cluster G_i, that is, the event E_{Gi} that*
  //*minimizes the distance to the other events in the cluster. Let S_j be the time series in*
  //*which the event E_{Gi} was found*
  $M = M \bigcup E_{Gi}$; //*add event E_{Gi} to the model*
  mark event E_{Gi} as examined;
  mark the most similar events to E_{Gi} as examined;
  //*from the cluster G_i obtain, for each time series $S_i \neq S_j$, the event E_p from S_i that is the //most similar to the representative event (E_{Gi}) output previously. Each E_p will be //represented in the model by the event E_{Gi} and therefore these E_p events will also be //discarded so as not to be considered in later iterations*
  $m = m - 1$;
end
return M as a model of the set S;

The most significant event clusters, that is, clusters containing events present in the highest number of time series, have to be identified in each iteration. Eq. (6) is used to calculate cluster significance:

$$SIGNF(G_i) = \frac{\#TS(G_i)}{n} \tag{6}$$

that is, cluster significance is given by the number of time series that have events in that cluster over the total number of time series n. Events that have already been examined are not taken into account to calculate the numerator. The most significant clusters are analyzed to output the events that are part of the model. To do this, the process of identifying the most significant cluster is repeated m times, outputting a representative event and marking both this representative and similar events in each time series as examined. With regard to the algorithm, note that:

(a) The algorithm is domain independent and it can be applied to any domain without any change. Fig. 4 shows a diagram of the structure of the proposed method that receives a set of time series S and generates a model M to represent S.
(b) The resulting representative event of the most significant cluster should not be taken into account again for the next iteration, and it is marked as an already examined event.
(c) As there may be several similar events in each time series, a cluster may contain more than one event from each time series. For this reason, clusters that are selected as being the most significant are not omitted in later iterations. The events already processed are marked as examined and will not be taken into account in future iterations.

Another important issue is the number of events making up the model. In this case, we have chosen the mode (m) of the number of events in the time series of S. This decision is based on the fact that, logically, a model that represents the original time series containing a typical number of events m should also have the same number of events m. The typical number of events in the time series of S may not be unimodally distributed. This could occur especially if there are not many time series in the set S. For non-unimodal distributions, we have opted to take the integer value closest to the mean of the number of events.

The same clustering technique is used in the clustering process as described in the time series comparison algorithm.

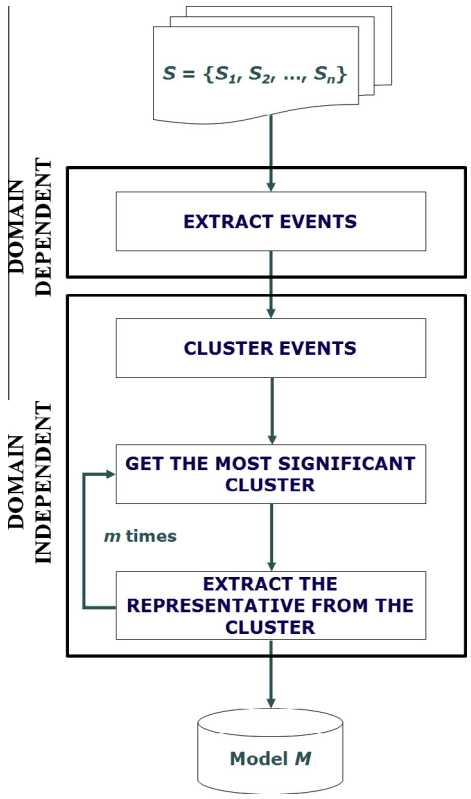


Fig. 4. Diagram of structure of the model generation method.

One last point to be considered is the distance between events used by the algorithm for clustering events, selecting representative events and discarding similar events. The *Manhattan* distance is used.

In order to create a reference model of n time series, it is necessary to extract all their events. To do this, all the time series have to be analyzed, meaning that the complexity of event extraction is $O(N{\cdot}n)$, where N is the number of timestamps in the series and n is the number of time series. The hierarchical clustering algorithm is run on the extracted events. This has a complexity of $O(q^2)$, where q is the number of events in all the time series. The typical number of events is then calculated, a process with a complexity of $O(n{\cdot}q)$. This process is less costly than clustering, as $n \leqslant q$. Finally, the extraction of the representative events of each significant cluster is a process of complexity $O(m{\cdot}k{\cdot}r)$, where m is the standard number of events, k is the number of clusters and r is the mean number of events in each cluster. Really, $k{\cdot}r$ is equal to q, meaning that the complexity of this process is $O(m{\cdot}q)$, less complex than clustering, as $m < q$. Therefore, if we obviate the previous events identification step, the global method has by extension of the clustering process a complexity of $O(q^2)$. In any case, square complexity with respect to the number of events is an acceptable complexity, as $N \gg q$.

Fig. 5 shows an example of the application of the proposed method to a set $S = \{S_1, S_2, S_3, S_4\}$ of four time series ($n = 4$). In this case, S_1 contains two events (E_{11} and E_{12}, S_2 contains two events (E_{21} and E_{22}), S_3 contains three events (E_{31}, E_{32} and E_{33}) and finally S_4 contains two events (E_{41} and E_{42}). Therefore, the typical number of events is two ($m = 2$). The extracted events are clustered into three different clusters (G_1, G_2 and G_3). Then, the most significant cluster is obtained. To do this, the significance of each cluster has to be calculated according to Eq. (6). In this case, cluster G_1 contains events present in three out of the four time series, cluster G_2 has events that appear in one out of the four time series and cluster G_3 has events present in four out of the four time series of S. Hence, the significance of G_1 is $SIGNF(G_1) = \frac{3}{4} = 0.75$, the significance of G_2 is $SIGNF(G_2) = \frac{1}{4} = 0.25$ and the significance of G_3 is $SIGNFG(G_3) = \frac{4}{4} = 1$. Therefore, the most significant cluster is G_3. In the next step, event E_{12} is extracted as the representative event of cluster G_3 because E_{12} is the event in G_3 that minimizes the distance to the other events in that cluster. Thus, event E_{12} is a characteristic event of S and will be part of the final model M. This process has to be repeated twice (because $m = 2$) to build the final model that consists of events E_{12} and E_{32}.

4. Case study: Experimentation in EEG

This section describes the application of the proposed framework in order to classify electroencephalographic time series.

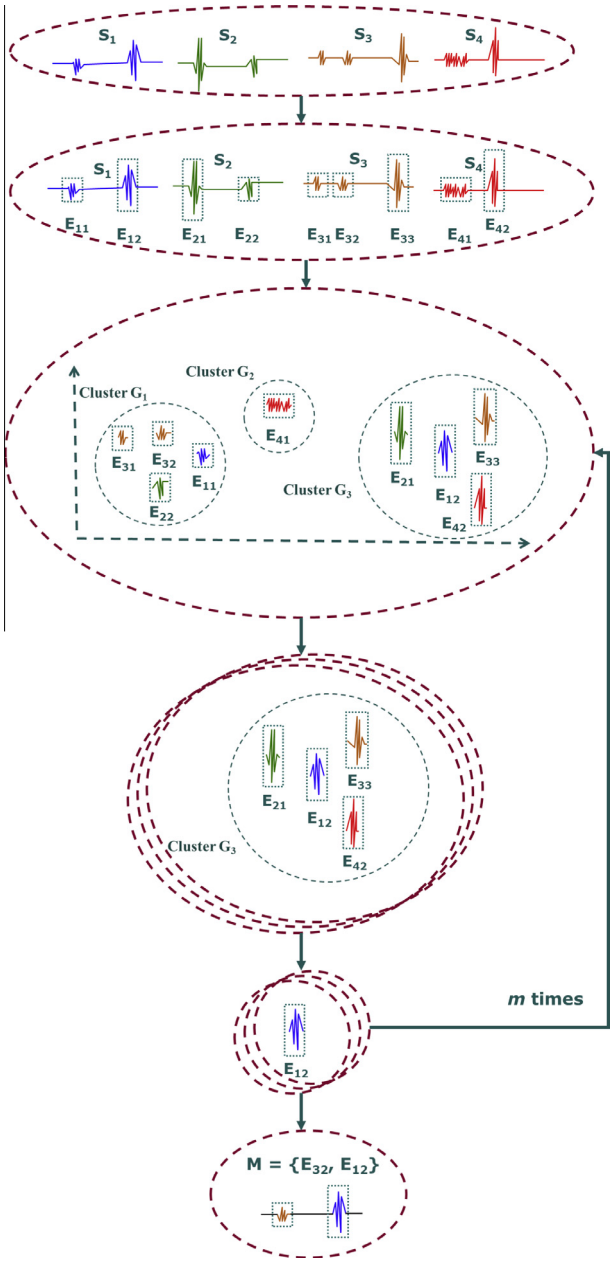


Fig. 5. Example of the application of the model generation method.

4.1. Domain

An EEG machine is a device used to create a picture of the electrical activity of the brain. It has been used for both medical diagnosis and neurobiological research, particularly for studying the mind-brain problem by searching neural correlates of mental activity. The essential components of an EEG machine include electrodes, amplifiers, a computer control module, and a display device. EEG machines are used for a variety of purposes. In medicine, they are used to diagnose such things as seizure disorders, head injuries, and brain tumors.

EEGs are time-series signals. Real-time information about the nature of these signals is often required for crucial decision making.

EEG analysis is a very useful technique for investigating the activity of the central nervous system. It provides information related to brain activity based on measurements of electrical recordings taken on the scalp of the subjects. The information obtained from an EEG is useful for inferences and studies about patient health and the effective treatment of many diseases [57].

The EEG signal is one of the most widely used signals in the field of biomedicine, as it provides a wealth of information to the experts. The relation of EEG signals to human movement and behavior has been extensively studied in past decades. EEG analysis has often been used to assist physicians in their diagnostic procedures, especially for differential disease diagnosis problems. Intelligent systems methods provide the opportunity to formalize medical knowledge and standardize several diagnostic procedures in specific domains of medicine and to store them in computer systems [58].

All EEG signals considered in this research were recorded with the same 128-channel amplifier system, using an average common reference. The data were digitized at 173.61 samples per second using 12-bit resolution. Band-pass filter settings were 0.53–40 Hz (12 dB/oct). Fig. 6 illustrates typical EEGs.

4.2. Data selection and recording

In our research we have used the publicly available data described in [59], where they compare dynamical properties of brain electrical activity from different recording regions and from different physiological and pathological brain states. Using the nonlinear prediction error and an estimate of an effective correlation dimension in combination with the method of iterative amplitude adjusted surrogate data, they analyze sets of EEG time series.

The complete data set consists of five sets (denoted A–E), each containing 100 time series. Sets *A* and *B* consisted of time series taken from surface EEG recordings that were carried out on five healthy volunteers. Volunteers were relaxed in an awake-state with eyes open (A) and eyes closed (B), respectively. Sets *C*, *D*, and *E* originated from the EEG archive of pre-surgical diagnosis. The duration of each time series is 24.6 s, and a total of 4097 timestamps were recorded a regular 6 ms intervals. Fig. 7 shows examples of five different sets of EEG signals taken from different subjects [60].

4.3. Event feature extraction

EEG devices generate time series that record scalp electrical activity (voltage) generated by brain structures.

EEG signals contain a series of waves characterized by their duration and amplitude. In EEG time series it is possible to identify certain types of special waves that are characteristic of some neurological pathologies, like epilepsy. Such waves are known as paroxysmal abnormalities and can be considered as events [61].

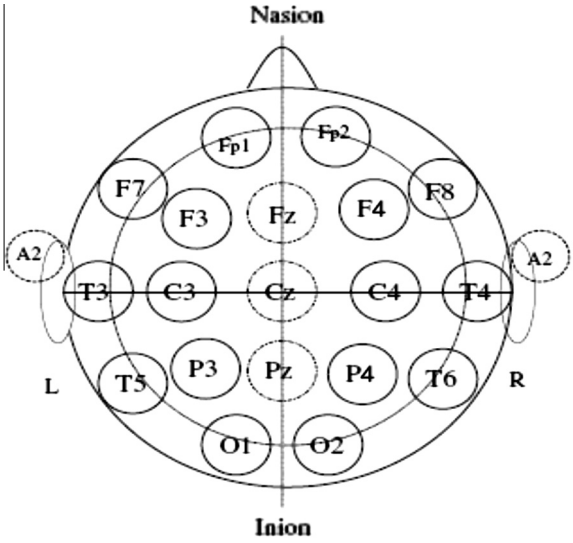


Fig. 6. The 10–20 international system of electrode placement.

During this research we have taken into account three kinds of events:

- Spike wave: a wave whose amplitude is relatively greater than the other waves in the signal. It has a duration of between 20 and 70 ms.
- Sharp wave: a wave whose amplitude is relatively greater than the other waves in the signal. It has a duration of between 70 and 200 ms. Fig. 8 shows an example of a sharp wave event.
- Spicule: a sharp wave with an abrupt change of polarity.

The features characterizing these events are as follows:

- Wave duration.
- Wave amplitude.
- Timestamp identifying when the event occurs within the time series.

An ad hoc method has been implemented to determine event features (as events extraction is domain dependent, an ad hoc events characteristics extraction method will have to be implemented to apply the framework proposed here to each particular domain).

We have used our event definition language to extract events from EEG time series. The definition of events proposed for the EEG domain is based on the identification of points where the polarity of the signal changes, as shown in Fig. 9. It is necessary then to identify points where there is a local maximum or minimum whose distance to the polarity change value is greater than a certain threshold (∂). That distance is the amplitude of the event. The duration of the wave is then calculated by analyzing the two intersections between the time series and the polarity change value line. Depending on its duration, the event is classified as a spike or a sharp wave, according to expert criteria. Finally, sharp waves that have an abrupt change of polarity are classified as spicules [61].

We consulted a panel of three domain experts in order to gather the above knowledge about events in the field of electroencephalography. These experts initially analyzed a total of 65 EEG time series containing a total of 392 events (249 spike waves, 123 sharp waves and 20 spicules) for training purposes. The number of events required to gather the target knowledge will vary depending on different factors: domain complexity, expert experience, etc., and the 392 events studied for this domain are considered sufficient.

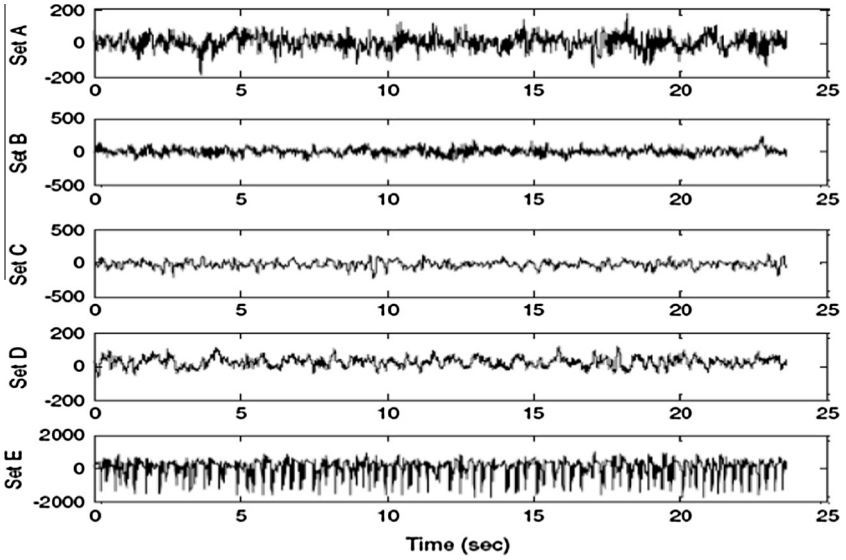


Fig. 7. Example of five different sets of EEG signals taken from different subjects.

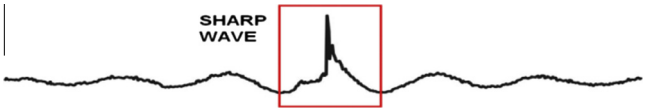


Fig. 8. Sharp wave event.

Based on the specified training, the events were defined and characterized symbolically using the proposed event definition language, and the result is shown below.

```
series EEGseries;
set minim
{
  // Filter and retain series minimums
  x in min(EEGseries)
  such that true
};
set maxim
{
  // Filter and retain series maximums
  x in max(EEGseries)
  such that true
};
event sharpWave
{
  peculiar_point in maxim,
  start in minim,
  end in minim
  such that  (EEGseries.value(peculiar_point) >
             mean (EEGseries) + 2.2 * stdev
             (EEGseries)) &&
             start == previous (peculiar_point,
             minim) &&
             end == next (peculiar_point, minim) &&
             (end - start)* _sampling period > 0.11
}; // Experts specified factors 2.2 and 0.11
event spikewave
{
  peculiar_point in maxim,
  start in minim,
  end in minim
  such that  (EEGseries.value(peculiar_point) >
             mean (EEGseries) + 2.2 * stdev
             (EEGseries)) &&
             start == previous (peculiar_point,
             minim) &&
             end == next (peculiar_point, minim) &&
             (end - start)* _sampling period < 0.11
}; // Experts specified factors 2.2 and 0.11
```

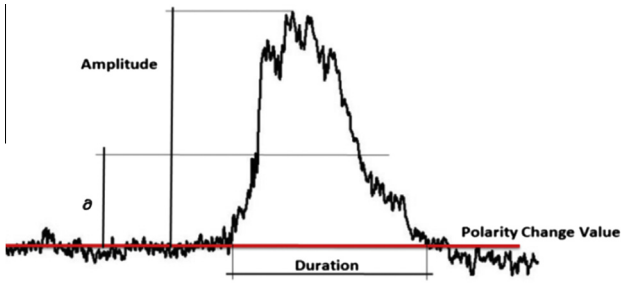


Fig. 9. Event taken from an EEG time series.

In the above definition, we find that the time series has only one dimension (one-dimensional time series). The spicules were obtained at a later stage by concatenating two previously located sharp waves.

The proposed definition is useful for identifying the three specific event types: spike wave, sharp wave and spicule, in this case. The main advantage is that, once the events have been defined, the proposed framework then automatically identifies events in electroencephalographic time series thanks to the proposed framework.

4.4. Classification of EEG time series. Discussion of results

EEG is now outmoded, having made way for other less invasive and more effective diagnostic tests. However, it is a branch of knowledge with many sample data published over many years and has proven to be very valuable from the statistical viewpoint thanks to the volume, unbiasedness and amplitude of the characterizations of the studied individuals. In this paper, we report a study of this diagnostic test as proof of concept for validating the proposed model against other well-known models.

In the study reported in this paper, we have conducted a series of experiments using *10-fold cross-validation* [62].

The test set used contains, as mentioned above, five data subsets (denoted *A*, *B*, *C*, *D* and *E*), each of which contains 100 EEG time series. This experiment has focused on sets *C* (healthy patients with open eyes) and *E* (epileptic patients during a seizure). It is precisely the wealth and availability of these data that has led us to explore this medical domain as a validation test of the proposed model. The goal of this evaluation is to determine how precise the classifications made using the framework are.

The first step is to identify events in the time series by means of our event identification language. This language is a consolidated and reliable tool that we validated at an earlier stage of our research [5].

Then, we had to create two reference models, one for each class ($M_{healthy}$ and $M_{epileptic}$). The first model ($M_{healthy}$) was created from a training set composed of 90 of the 100 time series in the set of healthy patients (*C*). The other 10 patients formed the test set. The second model ($M_{epileptic}$) was created from a training set composed of 90 of the 100 time series in the set of epileptic patients (*E*). The other 10 patients formed the test set. The patients from the test sets were selected at random. Table 6 shows the events of both models built with our framework and their characteristics.

Both created models were evaluated by checking whether $M_{healthy}$ adequately fits the group of healthy patients and whether $M_{epileptic}$ is representative of the epileptic patients group. To do this, we classified the 20 individuals in the test set according to their similarity to the two created models (this similarity value has been determined using the time series comparison method).

This entire process has been repeated 10 times, varying the training and test sets and obtaining the results reported in Table 7.

Other techniques have been used to classify EEG time series in the past. In [63], the author proposed a neural network system called AFINN (*Adaptive Fuzzy Inference Neural Network*) that is applied to the same EEG data set as we used in these experiments. AFINN, which is shown in Fig. 10, is a neural network composed of five layers. The first two layers, L_1 and L_2, are equivalent to the *IF* part of the fuzzy rules, whereas layers L_3 and L_4 contain information on the *THEN* part of the rules. Layer L_5 is the output layer.

Table 7 offers a comparison of the results of classifying individuals in the *C* (healthy) and *E* (epileptic) sets using the proposed knowledge discovery framework, the AFINN system and the multilayer perceptron. The perceptron achieved its best result using three layers with three neurons in the input layer, one neuron in the output layer and two neurons in the middle layer. A classical sigmoidal activation function and learning was by backpropagation using the mean square error as a measure of global cost.

Generally, all the methods return satisfactory results and correctly classify a high percentage of patients. We find that the rate of erroneous diagnoses (false-negatives) returned by the framework for epileptic patients is 4%, which is an improvement on the results of the multilayer perceptron but is worse than the AFINN system. On the other hand, the rate of false-positives (positive diagnoses in healthy patients) is 8%, which is worse than the other two methods, but is considered a reasonable or low-risk error rate. The confusion matrix shown in Table 8 summarizes these data.

Taking Table 8 as a starting point and analyzing the time series for epileptic patients incorrectly classified by our framework, the medical experts partnering this research found that they correspond to patients suffering a type of epilepsy characterized by minor (known as *petit mal*) or moderate seizures (known as complex partial seizures). A feature of these time series is that they have a lower incidence of events and are therefore hard to categorize using our approach, which explicitly focuses on processing perceptible events. The correctly classified time series, on the other hand, mostly correspond to patients that have a type of epilepsy characterized by severe or generalized seizures, known as tonic-clonic *grand mal* seizures (*International Statistical Classification of Diseases and Related Health Problems* (ICD), Diseases of the nervous system (G40.6)). These time series are characterized by a high incidence of events caused by a high nervous system electrical activity. Additionally, erroneous diagnoses by the AFINN model and the multilayer perceptron were, according to the medical experts, higher for this type of patients. Hence, we conjectured that our framework is better able to diagnose this type of disease than the other existing models, and we ran part two of the study to test this hypothesis.

In part 2 of the study, an independent expert selected patients from sets *A*, *B*, *D* and *E* suffering from *grand mal* epilepsy. The expert selected 100 cases that were absolutely certain to have the disease (classified as G40.6 according to ICD-10) and formed a new set of patients called *F*. Before using the data as model input, we ran a study of the sample to assure that the new sets *C* and *F* were not biased with respect to the studied quantitative and qualitative characteristics of each individual (this was unnecessary for the well-known sets *A*–*E* as they were test bench data that had been used and validated on many occasions). As set *F* was put together from different public data sets, this previous analysis is necessary to assure that the statistical study is reliable.

Sets *C* and *F* are of identical size, and a *Tukey HSD test* for post hoc comparisons to study possible significant differences in the sample and its homogeneous subsets is unnecessary. In its place, we ran an analysis of covariance (ANCOVA) to try to explain the qualitative variable epilepsy diagnosis = positive/negative in terms of the characterization of each individual in the study. This analysis aims to:

1. Check the occurrence of each study variable (age, sex, medical record, medical history, etc.) in each set (*C* and *F*) to assure that the sets are homogeneous with respect to the above variables, and the only statistically significant difference is the epilepsy diagnosis. Table 9 shows the result of the ANCOVA analysis analyzing each factor or study variable according to an orthogonal *sum of squares* (Type III) analysis. The *Pr* > *F* column indicates that the explanatory variables are not biased, as their values

Table 6
Reference models generated by the framework for EEG.

Model	#Event	Type	Duration (ms)	Amplitude	#Timestamp
$M_{healthy}$	1	Spike wave	35	80	2467
	2	Spike wave	23	78	45
	3	Spike wave	31	94	765
	4	Sharp wave	78	105	3012
$M_{epileptic}$	1	Spike wave	45	90	1246
	2	Spike wave	65	85	586
	3	Spike wave	53	e	2549
	4	Spike wave	55	103	3456
	5	Sharp wave	98	136	2045
	6	Sharp wave	102	148	3953

Table 7
Comparison of the EEG classification results of the different methods.

Patient type	#Patients	Knowledge discovery framework in time series (%)	AFINN (%)	Multilayer perceptron (%)
Epileptic	100	96	97.96	95.86
Healthy	100	92	98.12	94.98

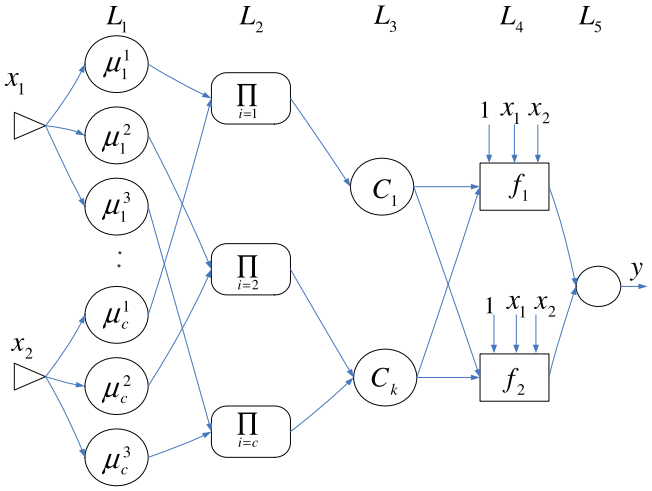


Fig. 10. Structure of AFINN neural network.

Table 8
Confusion matrix for EEG data.

	Epileptic	Healthy
Epileptic	96	4
Healthy	8	92

are much greater than 0.01 in all cases. Briefly, this means that individuals in the two sets, C and F, are comparable with respect to age, gender, record, etc.

2. Check the impact of each measured variable on patient diagnosis. Using a highly imprecise regression model for predicting the above variable, it is possible to assure that the sample is heterogeneous enough with respect to patient characteristics. This adds value to the model classification. In Table 10, we show the result of the regression model inferred by the ANCOVA analysis. The values R^2 and adjusted R^2 are very low, close to 0%. This means that the regression model is only able to classify 1.6% of patients (a value directly extracted from R^2. The impact of the explanatory variables on the final diagnosis is very low, and the sample of the sets *C* and *F* is heterogeneous. The model will make diagnoses whose effectiveness are unaffected by age, sex, medical record, etc.

We replicated the experiment conducted earlier on sets *C* and *E* using sets *C* and *F* on this new sample. Note that the events identified in this second experiment, composed of a subset of the time series, were the same as those identified in the first experiment on those series.

We created two models $M_{healthy}$ and $M_{epileptic}$ using the sample of 100 individuals of each type as earlier: 90 patients to build the model and 10 to test the model, selected at random.

Again, we repeated the experiment 10 times, varying the training and test sets. The results of this experiment are reported in Table 11, showing the mean number of well-classified elements in each set, the standard deviation and error, the 95% confidence intervals for the mean, and the minimum and maximum number of well-classified patients across 10 iterations.

Table 9
Analysis of covariance of type sum of squares (Type III).

Source	DF	Sum of squares	Mean squares	*F*	*Pr* > *F*
Age	2	0.635	0.317	0.874	0.754
Gender	1	0.478	0.478	0.954	0.861
Medical record	35	0.714	0.020	0.698	0.652
Medical history	20	0.221	0.011	0.983	0.873
Medication	12	0.324	0.027	0.897	0.791

Table 12 reports a comparison of the results of classifying individuals of sets *C* (healthy) and *F* (*grand mal* epileptics, classified in ICD-10 as G40.6) using the proposed knowledge discovery framework, AFINN system and multilayer perceptron.

Comparing the results of Table 7 (original experiment that includes all the healthy and epileptic patients) with the results in Tables 11 and 12 (second part of the experiment that includes the healthy patients and only the *grand mal* epilepsy patients), we find that, in the second case, the results of our framework are more accurate than the other analyzed proposals. This is because the patients with *grand mal* epilepsy exhibit marked and very isolated events in their time series, a condition to which our framework is especially well adapted.

Apart from the neural network-based approaches, the state of the art also describes special-purpose techniques for classifying time series. They include techniques based on the *k*-nearest neighbor algorithm. It is usual practice in this approach to use a measure of distance based on end-to-end differences among the series. The above experiments were repeated using this approach and resulting accuracy rates were close to 50%. This algorithm behaves like a random classification system. This is because, parts of the time series that are potentially of no interest to experts are considered in the analysis instead of being filtered out.

In search of the applicability of the nearest neighbor algorithm in EEG time series, we have used our special-purpose distance measure for time series with events. In this case, we have repeated the experiments described in Tables 7 and 12, achieving an accuracy rate much greater than 50% but less than 80% in the best case (for $k = 7$). This is because the nearest neighbors of each series to be classified do not necessarily contain the most representative events of each class. This circumstance suggests the need for an introspective analysis of time series events in such domains.

On the other hand, the *Shapelets*-based technique has the drawback of generating a single segment representing each class. That segment does not necessarily match any fragment of interest to the expert. Additionally, a reference model is generally composed of several representative segments (events) that are not necessarily

Table 10
Goodness of fit statistics.

Statistic	Value
Observations	200
Sum of weights	200
DF	135
R^2	0.016
Adjusted R^2	0.001
MSE	0.125
RMSE	0.353

Table 11
Descriptive statistics for the dependent variable diagnosis in the 10 EEG study iterations.

Set	*N*	Mean	Std. dev.	Std. error	95% Confidence interval for mean		Minimum	Maximum
					Lower bound	Upper bound		
Epileptic G40.6	100	99.863	1.491	0.087	99.547	99.926	98	100
Healthy	100	98.110	1.347	0.183	97.689	98.659	97	100

Table 12
Comparison of the classification of *grand mal* patients obtained by different methods.

Patient type	#Patients	Knowledge discovery framework in time series (%)	AFINN (%)	Multilayer perceptron (%)
Epileptic G40.6	100	99.86	96.26	96.61
Healthy	100	98.11	95.12	93

adjacent. On this ground, the average accuracy of this proposal in the experiments conducted on our data was at most 62%.

The accuracy of our proposal was greater than the other analyzed methods in the case of *grand mal* epilepsy. These are very important results as *grand mal* epilepsy is the most frequent type of severe epilepsy among the population. It is also more harmful to sufferers, as they tend to lose consciousness, collapse, suffer muscle spasms and may even die as a result of injuries from falls or oxygen starvation of the brain because they swallow their own tongue.

5. Case study: Experimentation in stabilometry

This section describes the application of the proposed framework for classifying stabilometric time series.

5.1. Domain

Stabilometry is the branch of medicine responsible for examining balance in human beings. Balance and dizziness disorders are probably two of the most common illnesses that physicians have to deal with. Around 30% of population suffers from some kind of dizziness disorder before the age of 65 [64].

Balance is examined using a device, called a posturograph. The patient stands on a platform and completes a series of tests. These tests have been designed to isolate the main sensorial, motor and biomechanical components that contribute to balance [60].

The posturograph generates a time-series signal, where the main information is normally confined to events.

5.2. Data selection and recording

A static Balance Master posturograph has been used throughout this research. In a static posturograph, the platform on which the patient stands does not move. The platform has four sensors, one at each of the four corners: right-front (*RF*), left-front (*LF*), right-rear (*RR*) and left-rear (*LR*). Each sensor records a datum every 10 ms during the test. This datum is sent to the computer connected to the posturograph. The datum is the intensity of the pressure that the patient is exerting on that sensor. Data are recorded as four-dimensional time series (one dimension for each sensor).

The Balance Master posturograph can be used to run a wide range of tests according to a predefined protocol. This research has focused on the Unilateral Stance (UNI) test that is the most useful for domain experts (physicians) in terms of output information. The UNI test aims to measure how well the patient is able to keep his or her balance when standing on one leg with either both eyes open or both eyes closed for 10 s. The UNI test generates time-series signals containing events, that is, regions of special interest for experts in the domain [64].

This research has been carried out on time series from a set of healthy patients, including both genders. In this paper, we have used 56 stabilometric time series with 1000 timestamps. The data set was divided into two classes according to patient gender: *Male* and *Female*. Of the 56 time series, 38 belong to male patients and 18, to female patients.

5.3. Feature extraction of events

The ideal situation for the UNI test would be for the patient not to wobble at all but to keep a steady stance throughout the test. According to the knowledge extracted from expert physicians, the interesting events in this test occur when the patient becomes unsteady, loses balance and puts the lifted leg down on the platform. This type of event is known in the domain as a *fall*, and there could be several *falls* throughout the same time series (the subject can lose and regain balance any number of times). The features characterizing falls are as follows: *Duration* of the fall; *Intensity* of the fall; and *Timestamp* at which the event occurs. When there is a fall, the respective sensors for the lifted leg will register the pressure increase. We have used our event definition language to extract events from stabilometric time series. Events in the stabilometric domain are defined by calculating the mode of the time series related to the lifted leg. This value represents the balance value as shown in Fig. 11. It is necessary to identify points where there is a local maximum whose distance to the balance value is greater than a threshold (∂). That distance is precisely the intensity of the fall. The duration of the fall is then calculated by analyzing the two intersections between the time series and the balance value line. The timestamp at which the event starts is the first intersection between the time series and the balance value line [60].

We consulted a panel of three experts in the domain in order to gather the above knowledge about the events in the field of stabilometry. The experts stated the importance of the morphological characterization of the events (defined by duration and amplitude), as well as the time of event occurrence. These experts analyzed 60 time series containing a total of 307 events. The number of events necessary for gathering the target knowledge will vary depending on several factors, like domain complexity, expert experience, etc., and 307 are considered sufficient for this domain.

Based on the stated training, the events were defined and characterized symbolically using the proposed event definition language, and the results are shown below.

```
// Dimensions
series lf;
series lr;
series rf;
series rr;
set candl
{
  x in lf
  such that
    (x in max(lr))
    && (∃ y in max(lf) such that abs(x-y) < TIME AXIS
  THR)
    && (∃ z in min(rf) such that abs(x-z) < TIME AXIS
  THR)
    && (∃ w in min(rr) such that abs(x-w) < TIME AXIS
  THR)
}; // The TIME AXIS THR is elicited from the expert
set cand2
{
  y in candl
  such that
    ((lf.value(y) + lr.value(y)) − (mod(lf) +
  mod(lr))) > INTENSITY THR
}; // The INTENSITY THR is elicited from the expert
set intersec
{
  z in lf
  such that
    lf.value(z) == mod(lf)
};
event fall
{
  peculiar_point in cand2,
  start in intersec,
  end in intersec
  such that
    (previous(peculiar_point,intersec) == start)
    && (next(peculiar_point,intersec) == end)
};
```

In the above definition (explained in detail in [5]), we find that the time series contains four dimensions, one for each of the four sensors used by the stabilometric platform to record time series. We observe that the proposed language is powerful enough to establish conditions that interrelate the different dimensions and is, therefore, able to identify the events of this type of multidimensional time series.

The proposed definition is useful for identifying a specific type of event: *fall* in this case. The main advantage is that, once the event has been defined, the proposed framework will automatically identify *falls* in stabilometric time series.

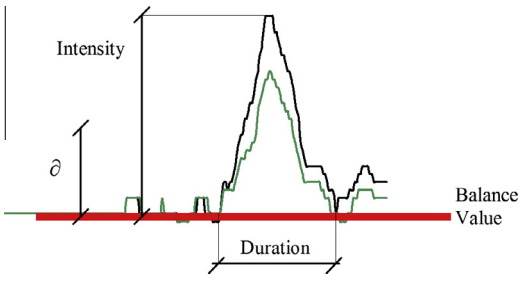


Fig. 11. Fall event taken from a stabilometric time series.

5.4. Classification of stabilometric time series. Discussion of results

We applied our framework to a second domain, stabilometry, in order to confirm the promising results obtained in the first case study. This is another domain where time series contain events.

In this case too, a series of experiments were conducted with the 70 stabilometric times series considered, divided into two groups: *Male* (40 time series) and *Female* (30 time series). According to experts, the balance of men is different to the balance of women, and this is one of the reasons that led us to use this data set to test the classification capability of our framework. Another reason for using this data set is that it has also been studied previously by other authors who applied the AFINN neural network for classification purposes [60]. Stabilometric time series classification is potentially very useful in sports medicine for, for example, recommending sports for young athletes, diagnosing possible balance-related diseases, etc.

For validation purposes, we first identified the events in the times series by means of our event identification language. This language proved to be a reliable tool for identifying stabilometric events. In [5], we describe the language evaluation process where the system was applied to 1620 stabilometric time series of real patients and was found to have a reliability rate of 98%. During the validation, we asked the panel of stabilometric experts to identify the events in those series and we then applied our technique to do the same thing. For each time series, we have measured the accuracy of our proposal using Eq. (1) that measures the degree of similarity (*SIM_Exp_Lang*) between the number of events identified by the expert ($\#Ev_{Exp}$) and by our language ($\#Ev_{Lang}$). Note that this formula offers a normalized result in the interval [0,1], where 1 indicates a total coincidence between the number of events identified by the expert and by the language. The worst case is when the expert locates events in a series and our system does not identify any:

$$SIM\_Exp\_Lang = 1 - \frac{|\#Ev_{Exp} - \#Ev_{Lang}|}{\#Ev_{Exp}} \tag{7}$$

From a global analysis of the 1620 time series, we found that there is good match between the experts (who identified 8618 in the time series) and the proposed language (which identified 9137), as shown by the mean similarity, which is greater than 98%, between the expert and language (Table 13). This value is very close to the ideal [5].

After identifying the events, we built two reference models for our study, one for each class (M_{male} and M_{female}). Table 14 shows the events (*falls*) present in the two models built using our framework and their characteristics.

As in the EEG domain, we applied the *10-fold cross-validation* technique. In each of the 10 iterations, we used four time series from the *Male* group as a test set (a total of 40 classifications) and the other 36 were used as a training set. On the other hand, three times series from the *Female* group formed the test set in each of the 10 iterations (a total of 30 classifications), whereas the other 27 were part of the training set. Table 15 compares the results with the AFINN neural network and the multilayer perceptron on the same data set. The perceptron configuration was the same as described for the previous domain.

Table 13
Global results of the application of the events definition language to the stabilometric domain.

#Series	$\overline{\#Ev_{Exp}}$	$\overline{\#Ev_{Lang}}$	$\overline{SIM\_Exp\_Lang}$
1620	5.32	5.64	0.981

Table 14
Reference models built by the framework for stabilometry.

Model	#Event	Duration (ms)	Intensity	#Timestamp
M_{male}	1	945	104	423
	2	567	95	693
	3	1453	127	158
M_{female}	1	655	99	620
	2	892	102	246

Table 15
Comparison of the stabilometric classification results.

Patient type	#Patients	Knowledge discovery framework in time series (%)	AFINN (%)	Multilayer perceptron (%)
Male	40	97.5	94.3	92.5
Female	30	96.7	95	93.3

Our framework was more accurate than the other two analyzed proposals. The confusion matrix shown in Table 16 shows that there are two time series incorrectly classified by our framework. These time series were contrasted with experts in the stabilometric domain, confirming that they can be considered as outlier time series in their class.

As in the first case study, the classification performance of the existing methods of time series classification in the simulations was more or less random. The greatest accuracy for k-nearest neighbors (for $k = 5$) and for *Shapelets* was less than 75% and around 55%, respectively.

With respect to neural network-based methods, the experiments confirm that our proposal has the following advantages over classical learning approaches:

- Our proposal is more accurate than classical approaches in cases where the time series contain few isolated events. Precisely, this is the open problem described in the introduction and that our proposal aims to solve.

Table 16
Confusion matrix for stabilometric data.

	Male	Female
Male	39	1
Female	1	29

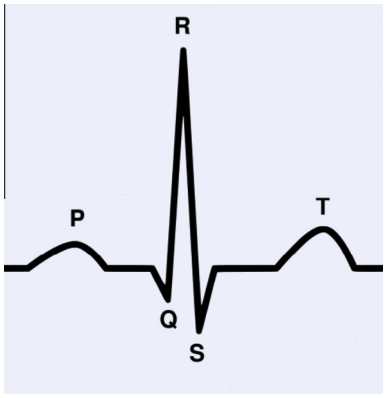


Fig. 12. Tracing of part of an electrocardiogram, showing the P and T waves and the QRS complex.

- The proposed framework not only offers good time series classification results but also generates reference models that can be interpreted by experts. These models are useful in the decision making process and help to find out more about the domain in question. Classical approaches are useful for classification purposes, but offer no such models.
- The models generated with our proposal can be used not only for classification but for other many data mining tasks, like, for example, outlier detection, clustering, etc.

6. Applicability of the proposed framework

In this paper we propose a data mining framework for time series containing events. In particular, our framework can be used to classify time series.

Assuming that the information of interest in the time series under analysis is concentrated in certain regions of interest (events), there is no other framework-specific constraint on its applicability.

There are several general aspects related to the applicability of our proposal:

- With respect to the need to consult application domain experts, the only domain-dependent part of our framework is the definition of events, which are, however, easy to define thanks to the proposed language, as the experience of the medical experts that participated in this research showed. The remainder of the framework is domain independent.
- With respect to the number of events in the time series, the experiments conducted appear to suggest that our framework performs well irrespective of the number of events. Performance was found to be more accurate in time series with not many,[1] prominent[2] events. Although length is not a factor limiting the applicability of our framework, it does bear a relationship to the number of events (the number of events will, in all logic, be smaller for shorter lengths). The results for time series with not many, prominent events are better because, if there are a lot of events, some events are more likely to be overlooked during identification (according to the knowledge elicited from experts). In such cases, some events are not considered as events because they do not meet the minimum conditions. However, some such events are borderline. The use of soft computing techniques could help to solve this problem, which we will address in future research.
- With respect to time series dimensionality, the experiments confirm that the framework is applicable to both one-dimensional (EEG) and multidimensional (stabilometry) time series.
- With respect to the possible periodicity of the time series events, there is no limitation regarding the applicability of our framework. The experts identify the conditions characterizing events and our system locates the events irrespective of whether or not they are periodic. A characteristic identifying the number of occurrences of each event in a periodic series of events could be incorporated in order to enrich the events. The framework would not need to be altered because of its generality and flexibility with respect to the number of events (as many as necessary depending on the domain).

According to the above, the proposed framework can be applied in many areas, both within and outside medicine. In the medical

[1] According to our experiments, when the length of the events is less than 30% of the total length of the time series.

[2] According to our experiments, when the amplitude or intensity of the event is greater than 25% of the total range of the possible time series values.

domain, it could be applied to other types of time series like, for example, electrocardiograms that contain periodic events. An electrocardiogram is the tracing output by an electrocardiograph measuring the electrical activity of the heart. The standard tracing of an electrocardiogram recording a normal heartbeat is a periodical sequence of a *P* wave, a *QRS* complex and a *T* wave (see Fig. 12). These are precisely the three event types to be located.

The aim of the *P* and *T* waves is to identify relatively low maximums. The event starts at the previous and ends at the next intersection of the time series with the mode. On the other hand, the aim in the case of the *QRS* complex is to find very high maximums (*R*), which are preceded by a relatively low minimum (*Q*) and followed by a relatively low minimum (*S*). A possible definition of events for this domain would be as follows.

```
// Time series definition
series cardio;
// Search points R, which are points where there is a
// local maximum at which the value of the time
   series is further than
// a certain threshold (THR1) from the mode of the
   series
set r
{
x in max(cardio)
such that
  abs(cardio.value(x) − mode(cardio)) > THR1
};
// Search points P and T (which will be
    differentiated
// later on), which are the points where there is a
    local maximum
// at which the value of the time series is further
    than a certain
// threshold (THR2) from the mode of the series
set pt
{
y in max(cardio)
such that
  abs(cardio.value(y) − mode(cardio)) > THR2
};
// Search points Q and ST (which will be
   differentiated
// later on), which are the points where there is a
   local minimum
// at which the value of the time series is further
   than a certain
// threshold (THR2) from the mode of the series
set qs
{
y in min(cardio)
such that
  abs(cardio.value(y) − mode(cardio)) > THR2
};
// Refine and extract points P from the set pt
// Points P are those points of pt such that the
   previous point of pt
// is at a sufficient temporal distance to be
   considered the point
// T of the previous heartbeat
set p
{
x in pt
such that
  !isfirst(x,pt) &&
   abs(cardio.value(x) −
  cardio.value(previous(x,pt))) > 2∗THR2
};
// Refine and extract points T from the set pt
// Points T are those points of pt such that the next
   point of pt
// is at a sufficient temporal distance to be
   considered the point
// P of the next heartbeat
set t
{
x in pt
such that
  !islast(x,pt) &&
    abs(cardio.value(x) −
  cardio.value(next(x,pt))) > 2∗THR2
};
// Refine and extract points Q from the set qs
// Points Q are the points of qs such that the next
    point of qs
// is close enough to be considered the point S of
    the same
// heartbeat
set q
{
x in qs
such that
  !islast(x,qs) &&
    abs(next(x,qs) - x) < UMB3
};
// Refine and extract points S from the set qs
// Points S are the points of qs such that the
    previous point qs
// is close enough to be considered the point Q of
    the same
// heartbeat
set s
{
x in qs
such that
  !isfirst(x,qs) &&
    abs(x − previous(x,qs)) < THR3
};
// Extract the intersections of the mode
set intersec
{
x in cardio
such that
  cardio.value(x) == mode(cardio)
};
// Extract events P, whose peculiar point is one of
    the
// points of set P and starts and ends at the
// previous and next intersections with the mode
event p
{
peculiar_point in p, start in intersec, end in
   intersec
such that
  previous (peculiar_point,intersec) == start
    && next(peculiar_point,intersec) == end
};
// Extract events T, whose peculiar point is one of
    the
```

```
// points of the set T and starts and ends at the
// the previous and next intersections with the
// mode
event t
{
peculiar_point in t, start in intersec, end in
  intersec
such that
  previous(peculiar_point, intersec) == start
    && next(peculiar_point, intersec) == end
};
// Extract events QRS, whose peculiar point is one of
  the
// points of R and starts at the previous point Q
// and ends at the next
// point S
event qrs
{
peculiar_point in r, start in q, end in s
such that
  previous(peculiar_point,q) == start
    && next(peculiar_point,s) == end
};
```

Three thresholds (THR_1, THR_2 and THR_3), whose value had to be determined by the domain expert, were used in this definition.

Based on this events definition, our framework could be used to characterize a particular heart complaint, provided that the electrocardiogram provided relevant information in this respect.

7. Conclusions and future work

We have developed a framework for classifying medical time series where key information is concentrated in specific regions of the series, called events, whereas the other regions of the series are irrelevant. The proposed schema is able to define events according to the knowledge extracted from domain experts.

The time series analysis techniques identified in the literature do not address this problem, as they analyze the whole time series. The main contribution of this research is the proposal of a framework that addresses the open problem of knowledge extraction from time series that contain events. Specifically, our framework provides functionality for: (1) comparing two time series that contain events, and (2) creating reference models from a set of time series. Elsewhere [5], we have reported an event definition language capable of identifying such events. Combining these three contributions, our framework has many potential uses in medical data mining, like classification of one-dimensional and multidimensional medical time series that contain events. Unlike other approaches that require major ad hoc low-level changes for each particular domain, the proposed time series analysis framework is, for the most part, generally applicable.

The method was evaluated on EEG and stabilometric time series, obtaining very satisfactory results, especially as regards the representativeness and accuracy of the reference models generated by the proposed method. Fig. 13 shows a screenshot of the implementation to test the framework in the stabilometric domain. Fig. 13 shows the reference model time series obtained for a group of patients. The features of the event highlighted on the time series chart are listed in the table above. The results confirmed that the proposed framework has a lot of potential as a classifier of EEG and stabilometric signals. The created reference models are potentially a very useful aid for medical experts diagnosing epileptic and balance-related disorders.

We have run experiments to compare our proposal with classical learning approaches based on neural networks. The results of our proposal were more accurate in both analyzed domains, and especially if there are not many events in the time series. Additionally, unlike the analyzed approaches, our proposal offers models that experts are able to interpret and can use for many other purposes apart from classification like, for example, outlier detection or time series clustering.

We have also conducted a comparative analysis of our proposal against other time series analysis techniques like, for example, time series classification techniques based on the nearest neighbor

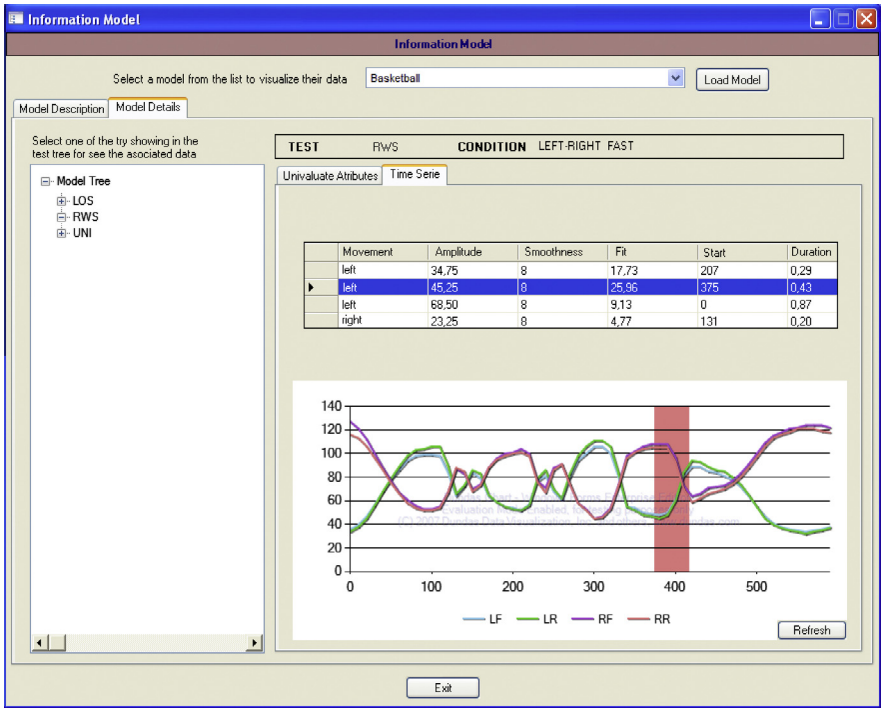


Fig. 13. Implementation of the proposed framework for the stabilometric domain.

algorithm or Shapelets. This analysis pinpoints the benefits of our framework and the need to use special-purpose techniques for extracting knowledge from time series that contain events.

One of the future research lines is concerned with event identification. One important aspect related to events definition is to determine the conditions that decide whether or not something is an event in a particular domain. In many domains, it is hard, if not impossible, to make such a distinction. The use of fuzzy logic looks to be both interesting and useful possibility for addressing this research line [60,65,66]. Using fuzzy logic, we would be able to build the concept of event *certainty* into the framework. This concept would indicate how sure we are about a region of the time series being an event [5]. The concept of certainty would lead to richer reference models where more importance would be attached to the more certain events. Additionally, when comparing two time series, the more certain events would carry more weight for determining the similarity between the time series in question.

Another research line on which we are working is the creation of an adequacy panel to enable experts to supervise the mechanisms for comparing time series and building reference models. It is expected that, thanks to this mechanism, we will be able to find possible discrepancies between the decisions made by our framework and determined by experts. The goal of this panel is to gather feedback from experts in order to adapt the proposed algorithms.

Finally, although the results compared with the other techniques are good, it is necessary to increase the number of reference domains in order to conclude that, statistically speaking, our method significantly outperforms the other analyzed proposals. To do this, we intend in the future to extend our proposal to a sufficient number n of domains (ideally, $n \geqslant 30$) on which carry out specific statistical tests to confirm this point.

Acknowledgments

This work was partially supported by the Spanish Ministry of Education and Science as part of the 2004–2007 National R&D&I Plan through the VIIP Project (DEP2005-00232-C03).

The authors would like to thank Ms. Rachel Elliott for translating this manuscript.

References

[1] Fayyad UM, Piatetsky-Shapiro G, Smyth P. From data mining to knowledge discovery: an overview. In: Fayyad UM, Piatetsky-Shapiro G, Smyth P, Uthurusamy R, editors. Advances in knowledge discovery and data mining. AAAI Press/The MIT Press; 1996. p. 1–34.
[2] Agrawal R, Faloutsos C, Swami A. Efficient similarity search in sequence databases, foundations of data organization and algorithms. Lect Notes Comput Sci 1993;730:69–84.
[3] Chan K, Fu AW. Efficient time series matching by wavelets. In: Proceedings of the 15th international conference on data engineering; 1999. p. 126–33.
[4] Povinelli R. Time series data mining: identifying temporal patterns for characterization and prediction of time series. PhD thesis, Milwaukee; 1999.
[5] Anguera A, Lara JA, Lizcano D, Martínez MA, Pazos J. Sensor-generated Time Series Events: a definition language. Sensors 2012;12(9):11811–52.
[6] Zhang W, Feng X, Bansal N. Detecting temporal patterns using RPS and SVM in the dynamic data systems. In: Proceedings of the IEEE international conference on information and automation; 2011. p. 209–14.
[7] Zhang W, Feng X. Pattern identification using reconstructed phase space and hidden Markov model. In: Proceedings of the 11th international conference on machine learning and applications; 2012. p. 374–89.
[8] Chaovalitwongse WA, Fan Y, Sachdeo RC. On the time series K-nearest neighbor classification of abnormal brain activity. IEEE Trans Syst Man Cybern – Part A: Syst Hum 2007;37(6):1005–16.
[9] Lee CL, Liu A, Chen W. Pattern discovery of fuzzy time series for financial prediction. IEEE Trans Knowledge Data Eng 2006;18(5):613–25.
[10] Yin J, Zhou D, Xie QA. A clustering algorithm for time series data. In: Proceedings of the 7th international conference on parallel and distributed computing, applications and technologies; 2006. p. 119–22.
[11] Tseng VS, Chen C, Hong T. Segmentation of time series by the clustering and genetic algorithms. In: Proceedings of the 6th IEEE international conference on data mining – workshops; 2006. p. 443–7.
[12] Kumar RP, Nagabhushan P, Chouakria-Douzal A. WaveSim and adaptive WaveSim transform for subsequence time-series clustering. In: Proceedings of the 9th international conference on information technology; 2006. p. 197–202.
[13] Perng C, Wang H, Zhang SR, Parker DS. Landmarks: a new model for similarity-based pattern querying in time series databases. In: Proceedings of the 16th international conference on data engineering; 2000. p. 33–42.
[14] Lara JA, Pérez A, Valente JP, López-Illescas A. Comparing time series through event clustering. 2nd International workshop on practical applications of computational biology and bioinformatics – advances in soft computing, vol. 49; 2009. p. 1–9.
[15] Rafiei D, Mendelzon A. Similarity-based queries for time series data. ACM special interest group on management of data; 1997. p. 13–25.
[16] Park S, Chu W, Yoon J, Hsu C. Efficient searches for similar subsequences of different lengths in sequence databases. In: Proceedings of the 16th international conference on data engineering; 2000. p. 23–32.
[17] Lee S, Chun S, Kim D, Lee J, Chung C. Similarity search for multidimensional data sequences. In: Proceedings of the 16th international conference on data engineering; 2000. p. 599–610.
[18] Wang Y, Zhou L, Feng J, Wang J, Liu Z. Mining complex time-series data by learning Markovian models. In: Proceedings of the 6th international conference on data mining; 2006. p. 1136–40.
[19] Jan L, Vasileios L, Qiang M, Lakaemper WR, Ratanamahatana CA, Keogh E. Partial elastic matching of time series. In: Proceedings of the 5th IEEE international conference on data mining; 2005.
[20] Dong X, Gu C, Wang Z. Research on shape-based time series similarity measure. In: Proceedings of the IEEE 5th international conference on machine learning and cybernetics; 2006. p. 1253–8.
[21] Chan PK, Mahoney MV. Modeling multiple time series for anomaly detection. In: Proceedings of the 5th IEEE international conference on data mining; 2005.
[22] Chen Z, Yang BR, Zhou FG, Li LN, Zhao YF. A new model for multiple time series based on data mining. In: International symposium on knowledge acquisition and modeling; 2008. p. 39–43.
[23] Caraça-Valente JP, López-Chavarrías I. Discovering similar patterns in time series. In: Proceedings of the 6th ACM international conference on knowledge discovery and data mining; 2000. p. 497–505.
[24] Srikant R, Rakesh R. Mining sequential patterns: generalizations and performance improvements, advances in database technology. Lect Notes Comput Sci 1996;1057:1–17.
[25] Ferreira PG, Azevedo PJ. Protein sequence classification through relevant sequence mining and Bayes classifiers. In: Progress in artificial intelligence; 2005. p. 236–47.
[26] Martínez HP, Yannakakis GN. Mining multimodal sequential patterns: a case study on affect detection. In: Proceedings of the 13th international conference on multimodal interfaces; 2011. p. 3–10.
[27] Lan Q, Ma C. A method of discovering patterns to predict specified events from financial time series. In: Proceedings of the 4th international conference on natural computation; 2008. p. 18–22.
[28] Guralnik V, Srivastava J. Event detection from time series data. In: Proceedings of the 5th ACM international conference on knowledge discovery and data mining; 1999. p. 33–42.
[29] Chung F-L, Fu T-C, Ng V, Luk R. An evolutionary approach to pattern-based time series segmentation. IEEE Trans Evol Comput 2004;8:471–89.
[30] Povinelli R. Identifying temporal patterns for characterization and prediction of financial time series events. In: Proceedings of the 1st international workshop on temporal, spatial, and spatio-temporal data mining; 2000. p. 46–61.
[31] Magnusson MS. Discovering hidden time patterns in behavior: T-patterns and their detection. Behav Res Methods Instrum Comput 2000;32(1):93–110.
[32] Granger CWJ. Investigating causal relations by econometric models and cross-spectral methods. Econometrica 1969;37(3):424–38.
[33] Shi X, Fan W, Zhang J, Yu PS. Discovering Shaker from evolving entities via cascading graph inference. In: Proceedings of the 17th ACM conference on knowledge discovery and data mining; 2011. pp. 1001–9.
[34] Ding H, Trajcevski G, Scheuermann P, Wang X, Keogh E. Querying and mining of time series data: experimental comparison of representations and distance measures. In: Proceedings of the 34th very large databases; 2008. p. 1542–52.
[35] Salzberg SL. On comparing classifiers: pitfalls to avoid and a recommended approach. Data Mining Knowledge Discov 1997;1:317–28.
[36] Xi X, Keogh E, Shelton C, Wei L, Ratanamahatana CA. Fast time series classification using numerosity reduction. In: Proceedings of the 23rd international conference on machine learning; 2006. p. 1033–40.
[37] Ye L, Keogh E. Time series shapelets: a new primitive for data mining. In: Proceedings of the 15th ACM international conference on knowledge discovery and data mining; 2009. p. 947–56.
[38] Baxt WG. Application of artificial neural networks to clinical medicine. Lancet 1995;346:1135–8.
[39] Jang J-SR. ANFIS: adaptive network based fuzzy inference system. IEEE Trans Syst Man Cybern 1993;23(3):665–85.
[40] Jang J-SR. Neuro-fuzzy modelling and control. Proc IEEE 1995;83(3):378–406.
[41] Sugeno M, Kang GT. Structure identification of fuzzy model. Fuzzy Sets Syst 1988;28:15–33.
[42] Takagi T, Sugeno M. Fuzzy identification of systems and its applications to modelling and control. IEEE Trans Syst Man Cybern 1985;15:116–32.
[43] Kovalerchuk B, Vityaev E, Ruiz JF. Consistent knowledge discovery in medical diagnosis. IEEE Eng Med Biol Mag 2000;19(4):26–37.

[44] Wiegerinck W, Kappen H, ter Braak E, Nijman M, Neijt J. Approximate inference for medical diagnosis. Pattern Recognit Lett 1999;20(11–13):1231–9.
[45] Walter D, Mohan C. ClaDia: a fuzzy classifier system for disease diagnosis. In: Proceedings of the congress on evolutionary computation; 2000. pp. 1429–35.
[46] Zahan S. A fuzzy approach to computer-assisted myocardial Ischemia diagnosis. Artif Intell Med 2001;21(1–2):271–5.
[47] Pena-Reyes CA, Sipper M. Designing breast cancer diagnostic via a hybrid fuzzy-genetic methodology. In: Proceedings of the 1999 IEEE int fuzzy systems conf; 1999. p. 135–9.
[48] Pattichis C, Schizas C, Middleton L. Neural network models in EMG diagnosis. IEEE Trans Biomed Eng 1995;42(5):486–96.
[49] Baader F, Calvanese D, McGuinness DL, Nardi D, Patel-Schneider PF. The description logic handbook: theory, implementation, applications. Cambridge (UK): Cambridge University Press; 2003.
[50] Anderson DR, Sweeney DJ, Williams TA. Quantitative methods for business. 7th ed. International Thomson Publishing; 1998.
[51] Scott G. Strategic planning for high-tech product development. Technol Anal Strateg Manage 2010;13(3):343–64.
[52] Jaccard P. Nouvelles recherches sur la distribution florale. Bull Soc Vaud Sci Nat 1908;44:223–70.
[53] Sorensen T. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons. Biol Skrifter, Kongelige Danske Videnskabernes Selskab 1957;5(4):1–34.
[54] Black PE. Manhattan distance. In: Dictionary of algorithms and data structures. PhD thesis, University of Westminster; 2009 [online].
[55] Kaufman L, Rousseeuw PJ. Finding groups in data: an introduction to cluster analysis. John Wiley & Sons; 1990.
[56] Yang P, Huang BA. Spectral clustering algorithm for outlier detection. International seminar on future information technology and management engineering; 2008. p. 33–6.
[57] Li J, Du Y, Zhao L. Sleep stage study with wavelet time-frequency analysis. In: International conference on neural networks and brain; 2005. pp. 872–5.
[58] Hanefeld F. Meeting report of the 4th congress of the European paediatric neurology society. Eur J Paediatr Neurol 2001;5(5):1–159.
[59] Andrzejak RG, Lehnertz K, Mormann F, Rieke C, David P, Elger CE. Indications of nonlinear deterministic and finite dimensional structures in time series of brain electrical activity: dependence on recording region and brain state. Phys Rev E Stat Nonlin Soft Matter Phys 2001;64:1–8.
[60] Jahankhani P, Lara JA, Pérez A, Valente JP. Adaptive fuzzy inference neural network system for EEG and stabilometry signals classification. In: Next generation data technologies for collective computational intelligence. Springer-Verlag; 2011. p. 329–55.
[61] Jahankhani P, Lara JA, Pérez A, Valente JP. Two different approaches of feature extraction for classifying the EEG signals, engineering applications of neural networks. IFIP advances in information and communication technology, vol. 363; 2011. p. 229–39.
[62] Kohavi R. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Proceedings of the 14th international joint conference on artificial intelligence; 1995. p. 1137–43.
[63] Jahankhani P. Development of a decision support framework for electroencephalography signals based on an adaptive fuzzy inference neural network system, PhD thesis, University of Westminster; 2009.
[64] Lara JA, Jahankhani P, Pérez A, Valente JP, Kodogiannis V. Classification of stabilometric time-series using an adaptive fuzzy inference neural network system. Artificial Intelligence and soft computing – lecture notes in computer science, vol. 6113; 2010. p. 635–42.
[65] Zadeh L. Fuzzy sets. Inform Control 1965;8:338–53.
[66] Zadeh L. Fuzzy sets, fuzzy logic, fuzzy systems. World Scientific Press; 1996.
[67] Liao TW. Clustering of time series data – a survey. Pattern Recognit 2005;38:1857–74.
[68] Shahar Y, Stein A, Musen MA. Knowledge acquisition for temporal abstraction. In: Proceedings of the AMIA annual fall symposium; 1996. p. 204–8.
[69] Stacey M, McGregor C. Temporal abstraction in intelligent clinical data analysis: a survey. Artif Intell Med 2007;39(1):1–24.
[70] Mörchen F, Ultsch A. Mining hierarchical temporal patterns in multivariate time series. In: Proceedings of the 27th annual German conference in artificial intelligence; 2004. p. 127–40.
[71] Catley C, Stratti H, McGregor C. Multi-dimensional temporal abstraction and data mining of medical time series data: trends and challenges. In: Proceedings of the 30th annual international conference of the IEEE engineering in medicine and biology society; 2008. p. 4322–5.
[72] Otero A, Félix P, Barro S. TRACE, a graphical tool for the acquisition and detection of signal patterns. Expert Syst Appl 2009;36(1):343–57.
[73] Batal I, Sacchi L, Bellazi R, Hauskrecht M. Multivariate time series classification with temporal abstractions. In: Proceedings of the 22nd international FLAIRS conference; 2009. p. 344–9.
[74] Hu B, Chen Y, Zakaria J, Ulanova L, Keogh E. Classification of multi-dimensional streaming time series by weighting each classifier's track record. In: Proceedings of the IEEE 13th international conference on data mining; 2013. pp. 281–90.
[75] Hariz M, Husain W, Abdul N. Data mining for medical systems: a review. In: Proceedings of the international conference on advances in computer and information technology; 2012. pp. 17–22.
[76] Chen J, Xing Y, Xi G, Chen J, Yi J, Zhao D, Wang J. A comparison of four data mining models: Bayes, neural network, SVM and decision trees in identifying syndromes in coronary heart disease. In: Proceedings of the 4th international symposium on neural networks: advances in neural networks; 2007. p. 1274–9.
[77] Strumbelj E, Bosnic Z, Kononenko I, Zakotnik B, Kuhar CG. Explanation and reliability of prediction models: the case of breast cancer recurrence. Knowledge Inform Syst 2010;24(2):305–24.
[78] Huang Y, McCullagh P, Black N, Harper R. Evaluation of outcome prediction for a clinical diabetes database. Knowledge exploration in life science informatics – lecture notes in computer science, vol. 3303; 2004. p. 181–90.
[79] Zhang S, Tjortjis C, Zeng X, Qiao H, Buchan I, Keane J. Comparing data mining methods with logistic regression in childhood obesity prediction. Inform Syst Front 2009;11:449–60.